Practice beyond technology when programming and mathematics teaching converge

Ana Fuentes Martinez
Practice beyond technology when programming and mathematics teaching converge

Ana Fuentes Martinez
Practice beyond technology when programming and mathematics teaching converge

Ana Fuentes Martínez 🌐 
Acknowledgments

Five years have passed since I started this academic adventure and there are many people I need to thank for it.

Along the way, my supervisors have guided and encouraged me, safe and steady. Thank you, Lars Svensson, for your great visions and inspiring advice, and thank you, Thomas Winman, for all the times you asked me to ‘write that down’ and elaborate my ideas. It is nothing less than a privilege to have you both to show the way.

To the teachers who altruistically granted me access to their everyday practices, their classrooms, and their plans, sincerely, thank you. Without your collaboration, your clever observations, and your pioneering work, this thesis would not have been possible. To all other mathematics teachers at my former school and in the extended professional community online, thank you for sharing your reflections and ideas on the topic of programming.

Many scholars have devoted their time to read and improve the multiple versions of this work and they all deserve my gratitude: Senior lecturer Peter Mozelius for his observant and challenging questions at the outset of my investigations; Professor Marcelo Milrad for contextualizing and sharpening the argument; Senior lecturer Lennart Rolandsson, for opening the gates of a systematic analysis of tactics upon completion of the licentiate thesis; and finally to Professor Åsa Mäkitalo, whose detailed scrutiny and precise remarks pushed the final draft towards both academic and professional relevance, your perspective on teaching and learning and your accurate insights made a great impact in the final steps.

Being a Ph.D. student opens the gates of a wide research community, in which knowledgeable and open-minded thinkers magnanimously share their insights and help each other. I had the blessing of being part of several spe-
cialized research environments: LINA (Learning in and for the New Working Life), GRADE (Swedish Graduate School for Digital Technologies in Education), and ExcITEd (Norwegian Center for Excellence in IT Education). I am grateful for all of you, junior and senior fellow researchers, for your support and constructive judgment, and for all the times you just dropped by, sent some words, or arranged a meet-up. Sara, Camilla, Linnéa, Ulf, Anh, Justyna, Ulrika, and Niklas, you will always be my best reason to cherish these five years.

I am fortunate to be surrounded not only by brilliant colleagues but also brilliant friends and family who made my journey joyful, the good moments better and the bad ones less so. To Carolina, thank you, among many other things, for initiating me into the intricacies of Michael de Certeau. To Hanna and Paula, whose true Gryffindor compass showed me the magic of being brave and kind when I really needed it, thank you. To my sister, thank you for making the 'Face with Tears of Joy'–emoji, the most popular on my phone. To my parents, for inculcating the values of education and perseverance, opening a world of possibilities, and trusting my choices, gracias de todo corazón.

I want to thank my dearest children for their unconditional support. Jens, thank you for your time with \LaTeX-tables and Python widgets, for sharing memes and cooking delicious meals. Lycke, thank you for helping me with the graphics, the transcriptions, and the proofreading, and for all the coffee-flavored conversations that make the evenings worth longing for. David, thank you for being there for swimming, reading, trip-planning, Pokémon hunting, and binge-watching Friends, those are the moments that count.

To Peter, my beloved husband, because you are always the first person I want to tell the good news, the scary dreams, the crazy ideas. For all your love and support, I am infinitely grateful.

The bits and bones of a life. The stones of her stream.

—Delia Owens, Where the Crawdads Sing
Populärvetenskaplig Sammanfattning

Titel: Didaktik bortom teknologi när programmering får plats i matematikundervisning

Nyckelord: Programmering; Matematik; Kursplan; Läroplan, Strategier, taktiker, Michel de Certeau

Denna avhandling undersöker hur programmering och matematikundervisning förenas i närvaro av en reviderad kursplan för matematik på gymnasiet. Fokus ligger på strategier som myndigheterna införde, hur lärare taktiskt navigerade de spännningar och motsägelser som uppstod i deras undervisning, och hur dessa taktiker senare konsoliderades i praktiken.

Två taktiska tillvägagångssätt blev tydliga när lärare började integrera programmering i matematik: Dual undervisning och sammanvänd programmering. Lärarens användning av duala undervisningsmetoder eller sammanvärd programmering var olika taktiker som formades av och som respons på villkoren i den nya läroplanen samt deras egna preferenser och syn på studenternas lärande. Dessa två taktiker avslöjar olika ontologiska åtaganden i förhållande till de strategier som representeras i styrdokumenten.

Successivt utvecklades de omgivande villkoren och gav upphov till ny undervisningspraxis. En andra omgång av intervjuer avslöjade hur senare reformer och nya insikter förändrade lärarnas praxis över tid. Lärarnas ursprungliga beslut både bekräftas och finslipas med nya erfarenheter och reflektioner. Andra taktiker behöver uppdateras, medan vissa kasseras. Processen mot en etablerad matematikundervisning med programmering ilustreras i relevanta kategorier där det framträder hur förändringen sker via bejakandet av programmering som praxis snarare än teknologi.
Abstract

Title: Practice beyond technology when programming and mathematics teaching converge

Keywords: Programming; Mathematics; Curriculum; Strategies, Tactics, Michel de Certeau

ISBN (print): 978-91-89325-64-7
ISBN (e-print): 978-91-89325-63-0

This thesis examines how computer programming and mathematics teaching converge in the presence of a revised mathematics curriculum for upper secondary education. The focus is on the stratified policy strategies deployed by the institutions; how teachers tactically navigated the tensions and contradictions that arose in their everyday teaching; and how these tactics later consolidated in practice.

The empirical data for the study consists of two iterations of individual interviews with nine mathematics teachers who were already proficient in programming at the onset of the reform. The teachers’ unit plans and other programming activities, were used as starting points for in-depth discussions about their professional practices. To gain a comprehensive understanding of the context, the author also examined relevant policy documents, including the mathematics curriculum, official guidelines, and a collection of programming exercises and demonstrations provided by the Agency for Education. Along with these documents, the official strategies were also informed by the explicit decisions and implicit outcomes surrounding the National Exams.

By analyzing teachers’ tactics and policy strategies, the thesis sheds light on the ways in which teachers adapted to the new curriculum and the chal-
lenges they faced in integrating programming into their mathematics instruction. This research aims to contribute to a critical understanding of the complex relationship between curriculum reforms, teacher practices, and the integration of programming in mathematics education.

When mathematics teachers started integrating computer programming into their subject, two tactical approaches became evident: dual teaching and interspersed programming. The teacher’s proclivity to implement dual teaching practices or interspersed programming are tactics shaped by and in response to the conditions of the new curriculum and their own preferences and views on student learning. These two tactics disclose different ontological commitments in relation to the strategies dictated by the curriculum and reflect a cardinal distinction between planning mathematics activities with elements of programming and planning programming activities with elements of mathematics.

Of relevance for teachers and curriculum designers is the understanding of (a) how the notion of programming and mathematics as separate subjects oversimplifies teachers’ actual integration practices, and (b) how the curricular choices made by policy can shape the teaching tactics adopted by educators.

Gradually, both the surrounding constraints and the reasons behind them evolved, raising new practices. The second iteration of interviews was designed to unveil the consequences of later curricular constraints and delve into the teachers’ practices as they change over time. Teachers’ initial resolutions, trials, and experiments with programming in mathematics are sometimes reinforced by means of perseverance and the teachers’ mature reflections on their past experiences. Other tactics need to be refined or updated and yet some are discarded. Along this distinction, relevant categories emerged that illustrate the processes behind consolidated practices in the presence of new technologies. Furthermore, the thesis provides a discussion on how this transition is characterized by acceptance of new practices rather than acceptance of new technologies.

Recognizing these aspects can guide educators and curriculum designers toward a better understanding of the complexities and nuances involved in integrating programming into mathematics education. This understanding can inform more effective teaching practices and curriculum development that support meaningful integration and promote students’ learning in mathematics with the help of programming.
Preface: From the case of programming to the paradigm of practice

When a five-year research project comes to an end and I look at the results neatly summarized in an abstract, it is easy to believe that the journey was as neat and straightforward as this final draft. It was far from being so. From how the topic came to be, to how the discussion made its place in the current academic dialogue, the road was a succession of distracting landscapes in monotonous drives, blind alleys, and unmarked forks. Likewise, there were lucky detours that made this thesis what it is. The path to knowledge is labyrinthine and its miscues should be acknowledged to give perspective and context to the text ahead, but also for the sake of those readers who might be pursuing a similar adventure. So, let us map it out!

Back in 2017, I had been working as a mathematics and programming teacher for more than a decade. At that time, we started hearing about an imminent curricular revision that would open for computer programming to make its way into mainstream syllabi. The contours of the reform were still vague, but I was thrilled to see that what I believed were essential skills were going to be made widely available for the upcoming generations. Soon after, it became clear that students would learn and use programming throughout their compulsory education and that programming should also be included in the mathematics courses in upper secondary school. That seemed to be my cue to take the opportunity and engage in what would be the beginning of a more-love-than-hate story.
Alongside my work as a high school teacher, I began to collaborate with the Swedish Agency for Education and traveled around the country introducing programming to mathematics teachers. It was an ambitious project organized in conferences and half-day workshops that offered many participants a first encounter with programming. The workshops aimed at introducing teachers to both coding skills and the synergy that knowing programming brought to mathematics learners. For me, being part of this effort, provided the possibility to meet not only many of the teachers who would bring programming to its realization but also some of the key people in the institutions driving the reform forward: curriculum designers, politicians, publishers, and other researchers in the field.

I also became involved in the local efforts to promote programming and develop a consistent progression scheme for the pupils from pre-school to high school. It was in the conjunction between the institutional efforts and the grassroots struggles that I started to realize the heterogeneity of the tensions that the reform both induced and unveiled.

In January 2019 I was offered a PhD position at University West within the newly established Swedish National Graduate School for Digital Technologies in Education (GRADE). There, I found a blooming academic environment tackling the questions raised by the digitalization strategy and its ramifications. My research proposal aimed at extending the knowledge on the relations between the mathematics curriculum and technology, particularly with regard to the consequences of using programming for teaching and learning mathematics. The university’s profile in Work Integrated Learning (WIL) provided a unique framework and a more precise direction to my research. The presentation of the problem was therefore concentrated on the relationship between working life and learning; in my case, Teacher Professional Development (TPD) for teachers that integrated programming in mathematics.

From the outset, my inquiry was pragmatic and exploratory, focusing on the practices of teachers who were implementing the reform and trying to learn from them possible ways of integrating programming and mathematics. What was intended to be a search for best practices led instead to a discussion on tactics and strategies (de Certeau, 1984) in which the activities in the classroom were interpreted in relation to the governmental directives, both with their own motives and their own goals; a response that sometimes suited and
sometimes antagonized with the intentions of the reform.

The insights from these initial investigations were tested, contrasted, and refined with data from subsequent empirical studies with the purpose of improving TPD in which programming and mathematics converged. However, during this process, several events marked the research project in ways that could hardly have been anticipated.

The first incident was yet another curricular amendment regarding mathematics in upper secondary school, this being the second one in less than three years. From June 2021, the mathematics content was rearranged within the different courses, with some parts being rendered obsolete and others reappearing after a ten-year-long break. Along those changes, the weight of programming was downplayed, in what could be a strategic move attending to the actual uptake in schools.

The second mayor event was the global medical emergency caused by the covid-19 pandemic, which profoundly changed the digitalization priorities in schools, relegating the initiatives for teacher training in computer programming in favor of more urgent matters of online learning.

On 24 February 2022, Russia invaded Ukraine, and an ongoing refugee crisis began in Europe. Swedish schools enrolled more than six thousand new pupils from Ukrainian refugee families under the Temporary Protection Directive \(^1\) (Skolverket, 2022b).

Later that year, sophisticated Generative Artificial Intelligence Tools were launched, allowing the general public access to chat-bots that could credibly type texts and even produce code on their own. This represents a radical change in educational premises that is prompting school districts to a number of quick solutions, such as AI-detecting functionalities and GPT teach-outs. Whether the technology is banned or embraced, teachers are again at the forefront of yet a game changer.

The aftermath of these all-unusual events highlighted the need for a more general approach to understanding the uptake of the reform. Was programming in mathematics just a fad that would be discarded as soon as any disruption came along? Would any new digital trend be enough to replace it? I sought answers to these questions in the tradition of Information Systems

\(^{1}\text{Council Directive 2001/55/EC. European Union directive providing for immediate, temporary protection for displaced people from outside the external border of the Union.}\)
and resorted therefore to the theoretical foundations of technology acceptance. Yet, the dissonance between the premises of new technologies in organizations and those of the reform nudged us to acknowledge the essence of programming in mathematics as a practice rather than a technology. A practice whose acceptance was not necessarily determined by the materiality of the programmable devices but also by the way mathematics was understood as a school subject, with its own purpose and its own traditions, forming the teacher identity of those who taught it.

The most tangible result of this journey lies ahead of you as a monograph organized in four parts. After being familiarized with the setting and the related research, the reader is invited to delve into the theoretical and methodological frameworks, harvest the results of empirical studies, and finally recognize the discourse above the pragmatics of the particular events.

But there are other outcomes, less ponderable but equally valuable, that also deserve some lines. I have had numerous enlightening experiences from courses, literature, colleagues’ feedback, editor reviews, and in the interactions with students and teachers. I had been able to spread my academic wings and find my own direction among the unexplored intersections of educational sciences, informatics, policy studies, and WIL. I have learned from each failure, each fall and each rise. I might have covered some of the bruises but the growth remains. After all, along with the thesis, another important outcome of a PhD is the student.

Conserva tus sueños, nunca sabes cuando te harán falta.

—Carlos Ruiz Zafón, La sombra del viento
# Table of Contents

Acknowledgments i

Populärvetenskaplig Sammanfattning iii

Abstract v

Preface: From the case of programming to the paradigm of practice vii

Part I 1

1 Introduction 3
   1.1 IS and WIL 5
   1.2 Purpose and research questions 6
   1.3 Contribution to knowledge 7
   1.4 Thesis Outline 8

2 Background 11
   2.1 The revision of the mathematics curriculum 12
   2.2 Curriculum 13
   2.3 Programming in mathematics 17

3 Teaching Programming 21
   3.1 Activities to learn programming 22
      3.1.1 Demonstrations 22
      3.1.2 Tinkering 23
      3.1.3 Debugging 26
   3.2 Progression in programming 28
Part II

4 Theoretical framework: Tactics, Strategies, and Practice

4.1 de Certeau and the theory of everyday practices

4.1.1 Everyday practices

4.1.2 Power and resistance

4.1.3 Tactics and strategies

4.1.4 Anti-disciplinary practices

4.1.5 Methodological notes

4.2 The Knowledge Quartet

4.3 The practice of teaching

5 The Empirical Study of Programming in Mathematics Education

5.1 Research Design

5.1.1 Methodological frame of reference

5.1.2 Research journey

5.2 Curricular documents and strategies

5.3 The voice of the teachers

5.3.1 Selection of participants

5.3.2 Interview design and implementation

5.3.3 Unit plans and other teaching materials

5.3.4 Classroom observations

5.4 Analysis of data

5.5 Abandoning the Knowledge Quartet

5.6 Mapping and finding tactics

5.7 Trust and cumulative knowledge

5.7.1 Ethical considerations

5.7.2 Addressing positionality

Part III

6 The strategic onset

6.1 Strategies behind the new curriculum
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.3  Tactics amidst change</td>
<td>147</td>
</tr>
<tr>
<td>11  Practice beyond tactics</td>
<td>151</td>
</tr>
<tr>
<td>11.1  Forging teaching practice</td>
<td>152</td>
</tr>
<tr>
<td>11.2  Practice Acceptance</td>
<td>155</td>
</tr>
<tr>
<td>12  Thesis Summary</td>
<td>159</td>
</tr>
<tr>
<td>12.1  Outcomes of the investigation</td>
<td>159</td>
</tr>
<tr>
<td>12.2  Restrictions and limiting factors</td>
<td>163</td>
</tr>
<tr>
<td>12.3  Further research</td>
<td>164</td>
</tr>
<tr>
<td>References</td>
<td>167</td>
</tr>
<tr>
<td>List of Tables</td>
<td>187</td>
</tr>
<tr>
<td>List of Figures</td>
<td>189</td>
</tr>
<tr>
<td>List of Quotations</td>
<td>191</td>
</tr>
<tr>
<td>Acronyms</td>
<td>195</td>
</tr>
<tr>
<td>Glossary</td>
<td>197</td>
</tr>
<tr>
<td>List of Appendices</td>
<td>199</td>
</tr>
</tbody>
</table>
Part I
Chapter 1

Introduction

In July 2018, computer programming entered the Swedish mathematics curriculum for Upper Secondary Education, a consequential step rooted in a view of programming as a significant digital competence that belongs in every student’s education. The measure reignited the discussions about the objectives, structures, policies, and pedagogical practices involved in educational reforms toward digitalization. In the halls of academe, scholars have long examined the complementary nature of mathematics and computer programming (eg. Papert, 1980; Barker et al., 1988), and many revived the ideas from the eighties along with a range of innovative technologies and pedagogical methods (Psycharis and Kallia, 2017). Learning programming is now often considered in the realm of computational thinking (eg. Weintrop et al., 2016; Mannila, Dagiene, et al., 2014; Kohen-Vacs et al., 2020), in an effort to explore far-reaching cognitive benefits above the practical applications of coding.

The expectations here are high. An early introduction to programming is anticipated to yield significant improvements on many levels, such as meeting the needs of the future labor market and reducing the gender gap in technology careers (Regeringskansliet, 2017a; Fuentes-Martinez et al., 2023). From the student perspective, there is some evidence of synergy effects when programming is part of the curriculum in other subjects (eg. Rich et al., 2020) and maybe even learning transfer, when the knowledge and skills from programming activities bring about achievements in other domains (eg. Bernardo and Morris, 1994; Lansiquot and Cabo, 2015).
CHAPTER 1. INTRODUCTION

The path to a successful reform is fraught with perils, though. Several reports indicate that mathematics teachers and school principals are concerned about the necessity of extensive training programs for teachers to learn programming themselves, and the inequalities that might arise during the implementation period (Mannila, Nordén, et al., 2018; Misfeldt et al., 2019). In 2020, a comprehensive survey from the National Union of Teachers in Sweden reported that a whopping 68 percent of the teachers in ISCED 3 felt ‘quite insecure’ or ‘very insecure’ about teaching programming (Lärarnas Riksförbund, 2020). Surprisingly, and despite the numerous training initiatives that had been put into place during the previous years, the problem had not been ameliorated. On the contrary, there were now twice as many mathematics teachers who felt anxious about programming compared to a similar survey conducted three years earlier (Larsson, 2017). The difference was that the latter had collected answers from mathematics teachers once the reform was already operational. By then, teachers had already been teaching according to the new subject plans, which was not yet the case in 2017. Integrating programming in mathematics might turn out to be more difficult than the teachers had previously estimated.

Even before the inclusion of computer programming, lesson time for covering the mathematics curriculum was already scarce and the teachers had seen their planning time being eroded with other chores (Nyroos, 2008). The addition of programming to the curriculum might have been seen with suspicious eyes by some teachers and praised by others. Consequently, scholars continue to argue for a critical approach to integrating programming into traditional mathematics teaching, focusing on the progression of programming knowledge required for effective instruction (eg. Zhang et al., 2020; Foerster, 2016). While aspects of teacher readiness and students’ equal access to programming should not be disregarded, the focus of this work will be on the actual practices and structures of the integration of mathematics and computer programming in the context of non-compulsory mathematics education. Bringing forward the lessons to be learned from mathematics teachers that are already including programming activities will be valuable not only for those designing the professional development initiatives for the teachers that want to learn programming but hopefully also in future curriculum implementations, to sidestep the pitfalls and align the resources with the expectations.
1.1 Intersecting knowledge fields

The thesis is written within the field of Information Systems (IS) with a specialization in Work Integrated Learning (WIL). The research in these fields concerns individuals learning in professional contexts where information technology is of consequence.

IS, also called informatics, is a discipline concerned with the study and application of information technology, computational systems, and the effective management of information. It involves the design, development, and implementation of digital technologies in a variety of domains, including healthcare, social sciences, education, and engineering (Friedman, 2013). Allen S. Lee, who was the Editor-in-Chief of the iconic journal *MIS Quarterly*, coined the well-quoted statement defending the distinct nature of Information Systems (IS) as an academic discipline. “Research in the Information Systems field examines more than just the technological system, or just the social system, or even the two side by side; in addition, it investigates the phenomena that emerge when the two interact” (Avison and Elliot, 2006, p. 5).

The unique focus of Information Systems is therefore the emerging socio-technical phenomena that occur when digital technologies are integrated into various domains. These phenomena can include changes in organizational processes, social dynamics, and even individual behavior. Researchers in IS seek to understand and optimize these interactions by studying the design, implementation, and use of digital technologies in various contexts. Ultimately, the goal of IS is to create effective and efficient solutions that enhance the way we manage and process information in our daily lives.

Introducing programming in mathematics curricula belongs to the realm of Information Systems in that it concerns the broader activity between people (teachers, students, etc), organizations (schools, government agencies, etc), and technology (the programmable devices, the algorithms and data structures that are taught). It too adds and builds upon the WIL perspective in that it emphasizes the learning opportunities derived from professionals’ experiences with programming in their teaching practice (Billett, 2020).
1.2 Purpose and research questions

It is generally understood that the decisions teachers make have an impact not only on student achievement but also on the overall success of larger interventions such as the present curricular reform. When achievement is at odds, many look for remedies in technology and economic resources, but the choices teachers constantly make in their classroom instruction is a significant factor that needs to be taken into account. Investigating this requires finding reliable evidence about how teachers adjust to the new curricular demands and the motives behind their positions. Ultimately, this knowledge provides emancipative tools and learning spaces for teachers to come up with creative implementations. For policymakers, the call is to consciously embrace this and other upcoming reforms in the ever-swinging pendulum of education, better equipped to predict and understand the consequences of educational reforms.

To critically analyze teachers’ everyday context with respect to the curricular and professional constraints that govern their activities, and the social actors connected to this environment, the thesis is anchored in three dimensions: the curricular changes, the teachers as curriculum agents, and the subject of mathematics with computer programming. The idea behind this holistic approach is to enable deeper analysis and a kaleidoscopic understanding of teachers’ practice by combining different perspectives pertinent to the area of research.

The main goal of the thesis is therefore to understand and analyze the implementation of programming in the ISCED 3 mathematics curriculum in Sweden from the perspective of teachers’ decisions on learning activities. In order to understand the teaching practices prompted by the curricular requirements, the investigation is framed within theoretical perspectives that are productive for explaining the subjective decisions surrounding learning activities. These phenomena are therefore explored in terms of tactics, strategies, and practice, emphasizing the autonomy of the teachers within the constraints of the curriculum.

Parting from Maxwell’s idea of a real world that exists independently of our thoughts and perspectives (Maxwell, 2012, p. vii), the investigation is grounded in a critical realist approach to knowledge, in the belief that our understanding does point to reality to some extent. More specifically, criti-
1.3. CONTRIBUTION TO KNOWLEDGE

cal realism acknowledges “the role of subjective knowledge of social actors in a given situation as well as the existence of independent structures that constrain and enable them to pursue certain actions in a particular setting” (Wynn Jr and Williams, 2012, p. 787). The reality to which we direct our understanding in this particular context refers to a new mathematics curriculum taking legal effect in Sweden from July 2018 (Skolverket, 2019a). It includes also the social actors—the teachers—responding to the new circumstances with some degree of autonomy.

The investigation asks therefore three questions:

**RQ1** How was the programming reform envisioned, deployed and developed?

**RQ2** How did mathematics teachers adapt their practices in response to the addition of computer programming to the curriculum?

**RQ3** What could constitute a sustained mathematics teaching practice in the presence of programming?

1.3 Contribution to knowledge

The outcomes of this thesis address the scarcity of research on teaching mathematics with programming from the point of view of teaching practice in relation to official directives. This can help policymakers to understand the conflicts between policy and practice but also open new views for teachers and teacher educators regarding the possibilities that the curriculum for Upper Secondary Education opens for using programming in mathematics. These results can facilitate research-based recommendations to better meet the expectations of future curriculum revisions, and more broadly, policies that aim at sustained changes in practice with new technologies.

The thesis provides also a new perspective on de Certeau’s theory of everyday practices by applying and developing the ideas on tactics and strategies onto the realm of teaching mathematics with programming. The results call attention to two distinct tactics—Dual and Interspersed—that can be used to analyze even other situations where teaching needs to address transectional content and methods. Furthermore, the discussion adds to the discourse on what constitutes professional practice in the presence of technology and how this practice evolves and stabilizes to navigate external constraints.
CHAPTER 1. INTRODUCTION

In addition to the aforementioned contributions, the thesis also offers a stratified analysis of institutional strategies that provide a novel framework for interpreting policy intervention. The three meta-categories in this structure, —Regulative, Facilitative, and Mitigative—, traverse the visibility dimension and add explanatory quality to the narrative on strategies.

Finally, the thesis proposes a conceptualization of why programming in mathematics should be understood as a practice acceptance rather than a technology acceptance. By shifting the focus of acceptance towards practice, the research highlights the importance of embracing programming as an integral part of the teaching and learning process in mathematics, rather than merely accepting it as a technological tool. This conceptualization underscores the significance of integrating programming within the broader pedagogical practices and recognizes its potential to enhance mathematical understanding and problem-solving abilities.

The main contributions can be therefore outlined in the following goals:

• A map over the curricular reform in order to identify the key elements directing its implementation in schools;

• An analysis of how teachers can understand, adopt, and shape the implementation of programming in their teaching practice;

• A new perspective on de Certeau’s theory of everyday practices applied to teaching mathematics with programming;

• A stratified framework to analyze strategies;

• Insights into the durable effects on practice with regard to programming in mathematics;

• A conceptualization of programming in mathematics as a practice acceptance rather than a technology acceptance.

1.4 Thesis Outline

The thesis is divided into four skeletal parts, I-IV, organizing the ideas from general to specific and back again. This means that the text starts with a broad description of the field and the theories, followed by the particularities of the
empirical observations to finally zoom out in search of a wider understanding beyond the case data.

Part I serves to contextualize the study. This chapter situates the thesis within the digitalization reform in Swedish Education and introduces the fields of curriculum sciences and programming in mathematics. It proceeds with a deeper description of the particularities of the 2018 and 2021 curriculum revisions and a review of related research addressing the implementation of programming in schools, focusing on mathematics teaching (chapter 2). Here I clarify the objectives and the ideas supporting the decision to boost digital skills by incorporating computer programming into existing subjects such as mathematics. The controversies in the research community are addressed delving into questions of legitimacy and leverage of the reform to position the thesis in the debate. Along with curriculum, chapter 3 explains the second foundational element to understand this thesis; the ideas and methods belonging to teaching and learning computer programming that will be necessary for the analysis of the empirical results.

Part II presents the theoretical and methodological frameworks that will guide the collection of data and justify the interpretation of the empirical findings. Michael de Certeau’s notions of strategies and tactics (de Certeau, 1984) are discussed in detail in chapter 4 to serve as performative tools in the following analyses. The research methods in chapter 5 include detailed descriptions of the setting, the data collection, and ethical considerations determining how the study was conducted.

Part III operationalizes the previous theories and concepts to understand how the new curriculum and its conditions for implementation shape teachers’ classroom practices. The power of the governed, a recurrent theme in de Certeau’s theory of everyday practices, gives rise to three interrelated analyses. Chapter 6 is devoted to the disclosure of the strategies, the institutionally sanctioned plans whose objectives comply with the views of the power institutions that enforce them. Tactics, on the other hand, are subversive everyday actions characterized by a self-defined target. Subsequently, the tactical responses to the strategies are the matter of chapter 7. The argument presented here implies that teachers engage in two types of tactics, Dual Teaching and Interspersed Programming that in different ways align with and diverge from the curricular strategies envisioned for mathematics education. Finally, chapter 8
elaborates on how these non-regulated actions of resistance become accepted practices rather than accepted technologies.

Part IV concludes the thesis with a discussion of the findings in relation to the research questions and the relevant literature. The discussion is structured into three chapters, each delving into distinct focal points. Chapter 9 explains the strategies driving the reform, providing a framework for further analyses of policy. The integration of programming in mathematics is discussed in chapter 10. The chapter analyzes the implications for the current curriculum implementation and discusses how dual teaching and interspersed programming offer principles to guide and sustain the incorporation of computer programming in mathematics. Chapter 11 resorts to a second body of theoretical premises regarding the traditional Technology Acceptance Model and offers a perspective of Practice Acceptance. The particular case of programming in mathematics is further extended to discuss practice acceptance when technologies bring about new habits. The last chapter (chapter 12) provides a summary of the main contributions and concluding remarks denoting limitations and suggestions for future research.
Chapter 2

Background

The Swedish education system is highly decentralized. The national curriculum is defined by the parliament and government, and its implementation is overseen by central authorities, municipalities, and independent institutions in accordance with the legislative framework. The majority of school budgets are funded by municipalities (Skollagen, 2010).

The Swedish National Agency for Education plays a crucial role in monitoring and assisting the local improvement of school quality. Its mission encompasses several key aspects, including setting goals and knowledge requirements, offering support for the development of preschools and schools, generating and sharing new knowledge for the benefit of specific groups, and engaging in effective communication to drive improvement (Skolverket, 2020a).

In this chapter, the particularities of the 2018 curriculum revision in Sweden are discussed and contextualized in order to understand how the curriculum made its way towards the mathematics classrooms (section 2.1). This background information is completed in section 2.2 with a short overview of different interpretations of curriculum and how the concept is understood in this work. The chapter ends with some remarks on the historical liaison between mathematics and programming with relevance to the scope of the thesis (section 2.3).
CHAPTER 2. BACKGROUND

2.1 The revision of the mathematics curriculum

Many countries are adjusting their education plans to develop students’ digital competencies. These reforms ultimately seek to adapt to changes in organizational and technological structures in the workplace as well as in research fields. As a response to the critiques questioning the adequacy of previous ICT curricula (eg. Wells, 2012; Furber and Nurse, 2012), computational science is now being incorporated into and even replacing many of the existing ICT initiatives. The long-standing international assessment of mathematics and science (TIMSS) recognizes mathematics as essential to computing technology and software development. Since 2023, TIMSS’s monitoring tests are fully digital, capitalizing on the benefits of computer-based assessments (Mullis, Martin, and von Davier, 2023). Allowing children to engage in computer programming in the scope of problem-solving activities is also seen as a means of developing a wider range of computational thinking skills (eg. Zhang et al., 2020; Brennan and Resnick, 2012).

While most countries have opted for separate courses in computer science or programming alongside the traditional curriculum, Sweden has chosen the interdisciplinary path with the explicit purpose of clarifying the overall mission of education in strengthening students’ digital skills. This means that computer programming is introduced to pupils within already established subjects, such as mathematics and technology (Skolverket, 2019a; Regeringskansliet, 2017a).

For Upper Secondary School (ISCED 3), the new national mathematics curriculum establishes that computer programming is to be incorporated in mathematics “by giving students methods for mathematical problem solving, including modeling of different situations”. Programming plays also an important role in introducing varied representations of mathematical phenomena, describing thinking processes, and illustrating abstract concepts (Skolverket, 2019c, p. 11).

The mathematics curriculum for ISCED 3 does not expect teachers to instruct students about how to program, nor does it dictate which kind of programming knowledge is necessary for each course. A separate course in computer programming shall therefore still be offered as an elective subject for

---

1Swedish: Strategier för matematisk problemlösning inklusive modellering av olika situationer, såväl med som utan digitala verktyg och programmering (Skolverket, 2017b, p. 10)
2.2. CURRICULUM

all ISCED 3 students (see Appendix 12.3 for a schematic description of the Swedish school system).

The programming reform was introduced in Sweden in 2018. The new requirements were enforced simultaneously from kindergarten to upper secondary school, which proved challenging in several respects. First, it required that a large part of the mathematics and technology teachers in the country learned computer programming in a very short time; a provision that necessarily naturally extended to pre-service teachers and teacher educators. Secondly, simultaneity also meant that the planned gradual progression in programming abstraction—from understanding stepwise instructions to visual block programming and ultimately to text programming—would not be available for most students in the first years after the reform. This rapid and uneven implementation behooved teachers to cater to pupils with dissimilar skills regarding computer programming (Fuentes-Martínez, 2019) and deprived students of equal educational opportunities.

Alongside TPD needs and disparity in students’ previous knowledge, the didactic issues of how and when to use programming in mathematics were not directly addressed in the policy documents which left a vacuum for pedagogues and textbook publishers to fill themselves. Therefore, the experiences of teachers who already feel comfortable with both mathematics and computer programming are seen as a valuable asset to learn about possible ways of merging learning activities that focalize the views of the curriculum.

2.2 Curriculum as a means of regulation

When setting out the scene for this thesis, the educational reform implemented in Sweden in 2018 became the context in which to study teachers’ enactment of the new curriculum. However, sorting out the essence of curriculum, and thus delimiting it among all the activities that can be identified as educational, has proven to be a difficult task.

The educational institutions that culture has produced try to balance individual teaching initiatives with standardization across local practices. Standards would correspond to what Whatmore et al. (2003) consider the ways by which cultural evolution suppresses itself to allow for fabricated items to persist in time, a conscious effort to maintain knowledge structures that would
otherwise ebb out. However conscious the efforts toward homogenization are, the content in the learning standards is not always deliberate, but a legacy of patches and incremental modifications over what once was considered core knowledge. Papert, who was both a mathematician and educator, meant that “A set of historical accidents […] determined the choice of certain mathematical topics as the mathematical baggage that citizens should carry (Papert, 1980, p. 51)”. Papert’s pioneering visions about programming as a learning vessel recognized the potential of computers to transform not only the learning process but also the curriculum.

Curricula stand out as the fundamental documents that uphold educational standardization and in them, the explicit views of what we value from our common past and what we expect from our future. Curricula are therefore ontological entities in education that at the same time enclosure the epistemological views of the system, the knowledge society considers essential to keep and develop, or as Pinar et al. put it, “what the older generation chooses to tell the younger generation (Pinar et al., 1995, p. 847)”. While this description is elusive in its operational aspects and might lack practical guidance for analysis, it successfully identifies two of its essential elements.

First, it points out that curricula are a matter of human choice, and as such, they can be chosen differently. Secondly, it makes the distinction between those who oblige under a curriculum and those who decide upon it, a cardinal dichotomy that will be explored further in terms of tactics and strategies in the upcoming chapters (section 4.1). The implication that curriculum in essence is an instrument of power becomes evident from the fact those who implement it and those who endure or delight in its idiosyncrasy are seldom the policymakers behind it.

More pragmatic accounts on the definition of curriculum usually connect with the subjects comprising a course of study, its academic content, and the knowledge and skills students are expected to learn (Kliebard, 1989). The description is evasive and used on different scales. Some authors refer to individual teachers’ curricula and imply the specific lessons, assignments, and materials a teacher uses to organize and teach a particular course, normally in the context of higher education (eg Null, 2016; Becuwe et al., 2017). At the policy level, the concept refers to the written documents that include the learning standards and learning objectives from an institutional perspective (Wiles,
2.2. CURRICULUM

The overall goal of the curriculum in this sense is to encourage greater curricular standardization and consistency across local actors, grade levels, subject areas, and courses but also “to mediate between unlimited wants and limited resources” (Kliebard, 1989, p. 2); a dynamic compromise to appease tensions and discrepancies among all the interests involved.

Three lines of curriculum theory can be distinguished in literature: design, implementation, and evaluation. Theories on curriculum design are primarily concerned with the content and instruction materials whereas implementation and evaluation theories involve the teachers and the students respectively. This presents a linear discussion in which the curriculum is treated as a finished product after its design is completed, ready to be delivered to schools for implementation, and subject to later assessment (Penuel et al., 2007; Romiszowski, 2016). Curriculum design, similarly to design concerning many other institutional policies, might take into account issues of implementation but is seldom engaged in facilitating self-evaluation and subsequent accommodation. The dynamic cyclical nature of modern product development becomes impractical for the large and authoritative process of modifying curriculum. Not only are the feedback mechanisms slow and sinuous but also a threat to the stability and dependability that is expected from institutional decisions (Van den Akker, 2007). This is not to say that curriculum designers will disregard outcomes in previous curricular reforms but measuring the success of the educational reform is not supported in the curriculum itself as a policy document (Fishman and Krajcik, 2003; Romiszowski, 2016).

A more comprehensive interpretation of curriculum than that of defined pedagogical guidelines handed over to teachers for implementation is offered by Voogt et al. (2016), who reformulate curriculum development within the scope of collaborative design. They mean that “curriculum change is not likely to prosper when teachers are merely viewed as practitioners who are expected to implement the plans of others (ibid., p. 122)”. However, Voogt et al. point out as well, that the effectiveness of such an active involvement relies on teachers themselves. In a study addressing the integration of ICT in teaching, the authors showed that for educational reforms to succeed, teachers need to believe that changing their practice is necessary, that their effort will convey that change, and that they are indeed able to promote and bring about the transformation needed (Becuwe et al., 2017). Those preconditions
are more likely to occur in locally adaptable curriculum materials whereas, in larger curricular amends at the governmental policy level, teacher mediation is scarce and subsidiary to juggernaut political powers (Bassey, 2018).

It is nevertheless worth noticing that educational reforms are not necessarily seen as an accomplished product delivered by central education agencies but as a process that needs to accommodate teachers’ resistance, negotiation, and compliance. Fraser and Bosanquet (2006) offer in this approach a slightly different view of curriculum development, in which evaluation is not set against the intentions of the original design but rather in comparison to the final expectations after the teachers appropriate the new learning standards. They incorporate the teacher in their interpretation of curriculum-as-process and by doing so, they allow for new locally rooted assessment methods.

Two arguments link this view of curriculum to de Certeau’s theories discussed later (see 4.1.3). First, from an axiological perspective, Kliebard (1989) concludes in his curriculum studies that “like educational theory in general, all the important problems of curriculum include a value component”. This value component is critical to the tactical maneuvers in which teachers engage. Secondly, understanding curriculum-as-process renders a more generative lens, one that allows us to recognize the productive power teachers have in their appropriation of curriculum, flourishing in the crevices of current policy.

Lindensjö and Lundgren mention three arenas in the process of implementing a new curriculum, from its initial formulation in the boards of education to its final realization involving the students and the contextual factors in the actual teaching situation (Lindensjö and Lundgren, 2000, pp. 171-177). In between, they identify a transformation or mediation arena, in which the teachers appropriate the purpose of the curriculum and select teaching activities to convey their own interpretation of the new guidelines. It is in this transformation phase that the curriculum is instantiated to account for the diversity of student groups and teacher competence and it is here where the teachers’ autonomous decisions are most significant to the final realization of the curriculum. By using the term ‘arena’, Lindensjö and Lundgren allow for a parallel development of the curriculum implementation, in which official clarifications, comments, and examples pertinent to the formulation arena can be
2.3. PROGRAMMING IN MATHEMATICS

TIMSS, a well established international assessment of mathematics, has recently published a similar curriculum model in the context of mathematics. They too recognize three arenas referred to as the intended curriculum, implemented curriculum, and attained curriculum (Mullis and Martin, 2019, p. 4). The intended curriculum represents the mathematics and science knowledge that students are expected to learn. These are the learning standards as defined by curriculum policies and publications of respective countries. The implemented curriculum pertains to how the educational system is structured and organized to facilitate this learning. Lastly, the attained curriculum reflects the actual knowledge and skills that students acquire and demonstrate as a result of a new mathematics curriculum. The division here is intended for evaluation purposes, allowing for comparisons among curricula in different countries and longitudinally within a country.

In this thesis, curriculum and learning standards are used interchangeably, meaning written descriptions of what students are expected to know and be able to do at a specific stage of their education. Curriculum is hereafter restricted to the educational regulations issued through state-managed boards with respect to the content and assessment in one subject. The empirical data and the results are concerned with the implemented curriculum in the mediation arena (Lindensjö and Lundgren, 2000). It is here where we can see in action the transformation of mathematics teaching once programming is included.

2.3 Programming in mathematics education

The essence of technology is using some phenomena for a purpose. In mathematics education, there is a long tradition of technological aids with the purpose of facilitating calculations, supporting cognitive functions, and more recently, finding hidden patterns in large data sets. (eg. Trouche, 2005; Séroul, 2000). From widely accepted mathematical formulas and numerical tables for standard functions to mechanical inventions such as the abacus or the slipstick, and later on, electronic devices, the resources available have been used to simplify and accelerate tedious processes and for the sake of advancing to more abstract and complex mathematics.
CHAPTER 2. BACKGROUND

The relation between programming and mathematics has been present since the early days of computer science. Broley et al. (2018) establish that computer programming has largely overtaken the practice of many professional mathematicians in both research and applied fields. However, for many years, the legitimacy and integration of newer technologies into the mathematical practice of schools remained marginal (Guin et al., 2005, p. 1).

Programming started as a branch of mathematics in higher education that soon grew too large and became a core subject in the field of computer science (Campbell-Kelly, 2018). Its wide range of applications and the availability of manageable programming environments made programming a part of many other higher education programs as well as curricula in pre-university courses (Campbell-Kelly, 2018; Rolandsson and Skogh, 2014). In these new scenes, the affordances and the technical aspects of programming were essential whereas the connection with mathematics became less prominent.

A survey about computer science in secondary education showed that the majority of countries did not expect programming to support mathematics, despite many authors claiming cognitive benefits for learning mathematics with programming (eg. Foerster, 2016; Putri, 2018; Papert, 1980). This is nevertheless a debated issue, with many pitfalls around the assumptions of what qualifies as mathematical knowledge and what are the transfer effects regarding higher-order reasoning. Whilst those factors are undeniably important, they call for a student-centered research framework, which falls outside the scope of this study.

Already in 1997, the Swedish National Agency for Education included technology in mathematics among the goals to work toward (Skolverket, 2000, English version). In that document, it was regulated that the school in its teaching of mathematics should aim at ensuring that pupils

[...:] develop their knowledge of how mathematics is used in information technology, as well as how information technology can be used for solving problems in order to observe mathematical relationships and to investigate mathematical models (ibid., p. 61).

By introducing the wording “computer programming” in the later amendment (Skolverket, 2017a), information technology was shaped into a very specific set of activities with its own idiosyncratic ways of operating and representing knowledge. In this curriculum, programming in mathematics is fur-
ther narrowed to be a digital tool for problem-solving. This limited view is nevertheless challenged in practice. For example, the study by Bergsten and Frejd (2019) demonstrated how pre-service teachers in a Swedish school used programming “for the purpose of generalising students’ conceptual knowledge in mathematics” (ibid., p. 941).

Over the years, many scholars have devoted their investigations to the role of programming in mathematics. Pea and Kurland (1984) studied the cognitive effects of such an endeavor and concluded with doubts about whether programming promoted mathematical rigor or even meaningful understanding of concepts. On the bright side, they too saw possibilities for programming to improve problem-solving skills in mathematics (ibid., pp. 159-160). In line with current ideas in the field of Work Integrated Learning (WIL), John Monaghan calls for employing computer programming to make school mathematics relevant to activities beyond mathematics classrooms and to connect with out-of-school mathematical practices (Monaghan et al., 2016, p. 333). This particular view was further explored in Fuentes-Martínez (2020), where the work experience of computer programming professionals was merged with mathematics teaching by means of pre-service practice. David Berlinski claimed that “mathematics, like physics, may yet become an empirical discipline, a place where things are discovered because they are seen\(^2\)”.

Undoubtedly, there are resurfacing hopes for a fruitful convergence of mathematics and programming in primary and secondary education but the pitfalls are still plenty, particularly regarding the shortage of teachers who are prepared to teach an integrated programming and mathematics curriculum.

\(^2\)Ground Zero: *The Pleasures of counting* (Berlinski, 1997)
Chapter 3

Teaching and Learning Programming

Programming is an essential knowledge not only in Computer Science and Software Engineering but in many other disciplines in which automation is necessary. Programming courses have been around since the first programmable devices became accessible to students in higher education institutions. This has created a fertile ground for research in learning programming that now encompasses a broader learner group; from young children learning the rudiments to teachers interested in the pedagogy of programming.

The empirical data that is presented later in this thesis is concentrated around the teachers’ practice when programming is included in mathematics (chapters 8 and 7). To better understand the activities that the teachers bring into play, this chapter examines processes and methods traditionally found in the teaching practice of computer programming.

Two aspects of programming education are particularly useful to explain how mathematics teachers incorporate programming: the form and the progression in the programming activities (sections 3.1 and 3.2). While there are indeed other contributing elements in the field of teaching and learning programming,—notional machines, assessment, devices, content, and context, to name a few,—those fall outside the purview of this thesis and are only considered succinctly.
3.1 Activities to learn programming

Teachers can facilitate learning text programming in many ways and it is at the core of their profession to select wisely among the available possibilities. This corresponds with what de Certeau refers to as types of operations within the modalities of action which will be explored in detail in section 4.1. In the context of teachers’ practice, these types of operations direct the pedagogical process rather than the topic although the topic of the lesson will significantly influence the suitability of the activity.

In this section, some key components of a programming teacher’s toolbox are described, to serve as a guide in understanding the learning activities that the teachers design for their students. The following account is therefore selected to align with the mathematics curriculum requirements for ISCED 3 in Sweden. Consequently, it concentrates on traditional text-based computer programming and leaves out other paradigms, such as unplugged programming or reactive computing (see Krishnamurthi and Fisler (2019) for an overview of different models of computation). The activity forms discussed here represent examples of the pedagogical palette in the programming teachers’ practice that are relevant to the premises, that is, newcomers to text-based programming. These different activities are often found, independently or combined, in the learning materials intended for computer programming.

3.1.1 Demonstrations

A common method to work with programming is to demonstrate the code. Demonstrations allow instructors to showcase programming concepts and techniques in action. It can be enacted by presenting ready-made coding examples or by programming live in front of the students (Rubin, 2013). Demonstrations can be used as teasers, illustrating only the outcomes of the program, or they can be step-by-step walkthroughs of the coding process.

Rubin (ibid.) defines live coding as “the process of designing and implementing a [coding] project in front of the class during lecture period.” By systematically typing, compiling, and testing code to solve example problems, the teacher shows the process in parallel with the concepts. Furthermore, the students can see how the teacher diagnoses and corrects mistakes, learning in real time from the error messages that programmers are bound to encounter.
Similarly, Barker et al. (2005) compare the live approach to the pedagogical methods common in fine arts, wherein the emphasis lies on the process rather than the ultimate outcome. The authors argue that showing missteps, “did not undermine the professors’ authority; instead, they suggested the reality of using technology; weird things happen […] and it is okay to not know it all (ibid., p. 424)”. Live programming provides a demonstration of the code as well as the necessary debugging techniques to solve different mistakes.

While demonstrations are largely a teacher-centered activity, many teachers involve their students by having them make predictions about the code. Reflecting on problem solutions that emphasize understanding the abstractions underlying programs appears to be associated with enhanced learning outcomes (Pirolli and Recker, 1994). Classroom discussions are orchestrated to facilitate learning, correct misconceptions, and suggest improvements. Seeing how the teacher handles errors in real-time helps demystify programming as something experts do, making it more accessible to all students (Rubin, 2013).

### 3.1.2 Tinkering

Tinkering in programming refers to “playful experimentation” in a permissive and responsive environment (Berland, 2016). This means that the activity encourages the students to make changes in the code of a given program where they can immediately see the results of their modifications as well as undo the steps with undesired outcomes. Tinkering allows students to explore and re-create with code autonomously, empowering them by making them feel in control of the system (ibid.). On the downside, tinkering can be time-consuming, compared with guided exercises, and there are concerns regarding gender differences in exploratory behaviors that can coerce the propensity to tinker (eg. Jones et al., 2000).

Tinkering by itself, what Felleisen et al. call “tinker-until-it-works philosophy” can also result in a shallow or equivocal understanding of the underlying computational model (Felleisen et al., 2004, p. 57). Students’ indiscriminate changes in the code may eventually yield a working program that risks leading them to attribute the success to the wrong instruction. Further coaching, where students are encouraged to explain their modifications and discuss the impact on the program’s behavior is expected to alleviate these problems.

Together with demonstrations, tinkering activities promote program com-
preparation prior to code generation. Program comprehension resembles passive vocabulary in the realm of language acquisition, in which understanding (passive knowledge) precedes speaking (active knowledge). “The conversion of passive knowledge into active knowledge is thought to be one of the main goals of language learning” assert linguists Miller and Ginsberg (1995, p. 305). Analogously, producing code in programming can be seen as the active counterpart of program comprehension. While this transition from passive knowledge to active coding is not necessarily an objective when programming is included in mathematics, it signifies the ability to apply comprehension and turn it into solutions to new problems, which demonstrates a higher cognitive level.

**Templates and scaffolding**

There are several ways to guide students through constructing programs, all sharing the intention of reducing the extraneous cognitive load. This refers to the load that arises from the instructional design and that does not directly support learning (Atkinson et al., 2000). The common rationale lies therefore in identifying which of the barriers associated with constructing programs can be lifted without stymieing learning.

One of these barriers arises from having to fill the gap between the initial problem state and the final goal. This involves searching and selecting among conceivable methods which can dominate the thought process (Pirolli and Recker, 1994). Worked examples offer step-by-step guides showing how to solve an existing problem. Similarly to demonstrations, worked examples provide students with an expert solution to a programming problem. The activity resides in replicating the steps from the worked example with a comparable solution to isomorphic exercises (Atkinson et al., 2000).

The process can be instantiated with varying levels of granularity. Some authors suggest breaking the problem down into sub-goals and adding labels that resonate with the function of fragments of code (what does this do?) (eg Margulieux, Morrison, et al., 2020). These labels can be either provided in advance or become a part of the learning activity if the students are asked to generate them (Margulieux, Guzdial, et al., 2012). Providing increasingly variable problem subtypes are other instructional approaches that researchers have shown to improve students’ learning from worked examples (eg Gray
3.1. ACTIVITIES TO LEARN PROGRAMMING

Further levels of student autonomy are reached through programming templates. Templates are program skeletons that serve as starting points or frameworks for students to build upon. They provide a program structure and guidance for completion, allowing for customization and expansion. Together with examples and explanations, the learner shall be able to produce programs that address a new problem.

Pirolli and Recker (1994) investigated how scaffolded examples could be designed to improve the acquisition of programming skills. Their premise was that to be able to produce programs, the students needed to connect the example material and their prior knowledge to the abstract terms of the new problem (ibid.). Scaffolding shall therefore combine the processes that the students already master with new information to facilitate the transfer of knowledge into new programming domains Figure 3.1. Hence, curriculum progression is embedded in the scaffolded activity; a decision about the order of topics to be learned and skills to be acquired that build upon each other.

![Figure 3.1: Transfer model of programming skills acquisition by means of scaffolded examples (adapted from Pirolli and Recker, 1994, p. 242)]
Scaffolded activities and templates guide students in understanding how different components in the program interact and help them add specific functionality to existing code. The purpose of this kind of exercises is to provide a structure that allows learners to identify similarities among coding problems. Comments and labels allow students to communicate with their peers and instructors more efficiently. The balance lies in designing materials that reduce the cognitive burden without yielding the students overconfident about their understanding of the procedure or too dependent on external guidance.

3.1.3 Debugging

Debugging describes the process of finding and correcting errors (bugs) in a program. It is a key component of software development but also a frustrating and time-consuming barrier to students who are learning computer programming. It is a challenging endeavor as it requires the application of many skills simultaneously. Students must understand the problem domain and the logic of the intended program, but they need also to know rudimentary programming concepts and be able to read and write instructions in the programming language of the code to be debugged (McCauley et al., 2008). Despite this complexity, students normally acquire debugging skills in parallel with other programming competencies such as learning about algorithms and data structures.

Debugging techniques can be taught by letting the students engage with erroneous code. In subsection 3.1.1, the teacher was identifying, analyzing, and resolving errors live. Also tinkering is likely to render errors from which to learn the necessary troubleshooting. Scaffolding activities can be designed to guide the debugging process, by providing an instance in which the code fails and exemplifying techniques to fix it.

Discussions on the debugging techniques employed during the exercise are used to articulate the different errors that can occur. Whalley et al. (2021) suggest encouraging the students to form a hypothesis about the source of the error and even provoking invalid outputs by testing boundary conditions to elicit information about the problem. (Murphy et al., 2010) propose group or pair debugging activities to promote reflection and meta-cognition whereas other authors advocate the use of flowcharts or decision diagrams as a way to guide the process and allow the learners to take deliberated steps toward
finding and fixing the errors (eg. Michaeli and Romeike, 2019).

Understanding the importance of debugging, Carver and Risinger (1987) designed a flowchart with which novice programmers could learn to effectively narrow their search for mistakes in the code (Figure 3.2). They postulated that to develop the general problem-solving skills associated with programming, children should be taught debugging explicitly rather than along other programming exercises where debugging risked being watered down to inconsequential trial-and-error. Their objections are akin to those of Felleisen et al. (2004) regarding tinkering (subsection 3.1.2) and corroborate once more the importance of intentional cognitive activities. Pirolli and Recker’s reflections on transfer in the domain of programming lead to similar conclusions with respect to debugging skills, which follow from strong mental models of the programming tasks (1994, p. 273).

![Figure 3.2: Decision diagram for debugging programs as proposed by Carver and Risinger (1987, p. 158)](image)

Debugging is a central part of programming that has far-reaching benefits in understanding code. Wang et al. (2022) argues that being proficient in code comprehension largely influences the students’ ability to identify reusable units and capitalize on open-source programs.
This section provided a curated selection of the schemes that are common in programming instruction. Partial in nature, it focused mainly on how the students interact with code and left out of scope important elements that programming teachers also include in their lessons. Because of the intention to align with the programming activities that are both relevant for mathematics and suitable for novice programmers, more demanding tasks have prudently been omitted. These might be theoretical approaches to determining the complexity of algorithms in terms of time or memory usage, or quality decisions such as robustness, generalizability, or efficiency of a program.

Along with choosing the activities to support the acquisition of basic programming skills, the teachers need to take into account the central topics in the subject and their prospective sophistication. The conventional learning progression in programming education for beginners is therefore reviewed in the following section.

### 3.2 Learning progression for programming

When discussing how the teachers plan their mathematics lessons with computer programming, a learning progression in mathematics contents is assumed to exist in the background. This progression is explicitly established in the curriculum (Skolverket, 2019b). Students in upper secondary school are expected to solve equations with one variable before they move on to systems of equations with two or more unknowns, and they learn about real numbers before complex numbers are introduced. With some minor variations, the learning trajectory in mathematics reflects an increasing level of abstraction and difficulty as well as some expectations of consistency within the different units of the syllabus. This common path is enacted consequently in textbooks and national exams and, to a certain extent, they even influence how other parallel subjects present their contents.

Biology students will need to know about probability and statistics to grasp the basics of population genetics and later on, they will be expected to handle differential equations to model ecological mutual relations that explain why sharks that eat a tuna population to near extinction, will then see their own shark population decline, too. In the same way, physics teachers might need their pupils to learn about trigonometry before they introduce forces in an
3.2. PROGRESSION IN PROGRAMMING

inclined plane and complex numbers will be a prerequisite to quantum mechanics.

For text-based programming, the major challenge is mapping syntax to behavior, because computer behavior is hard to see and control. To tackle this problem, a very similar knowledge ladder can be found in most introductory textbooks on the subject, particularly when the programming fundamentals are presented. It starts by presenting variables and sequentially-structured programs followed by conditional and repetition structures and later complex structures such as nested constructs, lists, classes, recursive functions, etc. (eg. Gluga et al., 2012; Bers, 2020; Mayer, 2013; Fuentes-Martínez, 2021).

Moreno-León et al (2017) constructed an analytical tool to assess progression in terms of coding and computational thinking skills. The rubrics were developed to assign complexity scores to programming solutions from young programmers working with block programming. Nevertheless, most of the instructions can be extrapolated to the text programming environments required in the ISCED 3. Table 3.1 replicates a fragment of the original table showing the Computational Thinking dimensions that are considered relevant for understanding the results of this thesis. The entries in light cursive font are generally not applicable to the kind of programming that mathematics teachers bring to their lessons.

This progression perspective is akin to the one in the syllabus proposal produced by the Swedish Agency for Education to guide the teacher training courses commissioned to universities (Statens Skolverk, 2017). The cognitive development implied in this progression concedes that in order to program conditional structures, you need to know about what a sequence of instructions does and that before diving into an aggregated data set it is advisable to understand how individual variables work. At this elementary level, there is however no external progression that would inform what specific programming knowledge is necessary prior to advancing in other subjects. This position is endorsed in Mørch and Kafai (2022), where the interviewee remarks on how students could learn programming in parallel physics with almost no previous programming experience. In this sense, computer programming resembles a tool, a skill that is necessary as a whole but can be improved in parallel with other subjects.

This instrumental view has fundamental repercussions not only on the
way programming is taught and learned but also on the assessment methods that thus are enabled in cross-curricular settings (Fuentes Martinez et al., 2023a), i.e. when programming is integrated into the curriculum of other subjects.

**Table 3.1:** Programming Proficiency Levels according to Dr. Scratch\(^1\) mastery scores (Adapted from Moreno-León et. al., 2017). The instructions in regular font can be extrapolated to text programming.

<table>
<thead>
<tr>
<th>CT dimension</th>
<th>Proficiency Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Basic</td>
</tr>
<tr>
<td>Logical Thinking</td>
<td>if</td>
</tr>
<tr>
<td>Data representation</td>
<td>modify object</td>
</tr>
<tr>
<td>Interactivity</td>
<td>green flag</td>
</tr>
<tr>
<td></td>
<td>system input</td>
</tr>
<tr>
<td>Flow Control</td>
<td>sequence of blocks</td>
</tr>
<tr>
<td>Abstraction</td>
<td>more than one script</td>
</tr>
<tr>
<td>Decomposition</td>
<td>more than one sprite</td>
</tr>
</tbody>
</table>

In the following chapters, the level of complexity for the different programming solutions is assessed according to this general progression. The scale is in no account absolute and thus it will be used loosely. The purpose of including a progression scale, albeit vague, is to understand whether certain programming exercises might be considered more advanced than others. Rather than presenting a clear-cut classification of the activities in order of cognitive sophistication, these proficiency levels are intended to determine whether a progression in programming difficulty might be an implicit part of the teacher’s tactics.

\(^{1}\)http://www.drscratch.org/
Part II
Chapter 4

Theoretical framework: Tactics, Strategies, and Practice

A unique empirical material on mathematics teachers’ everyday practices and their choices regarding teaching with programming was collected at two different points in time, first at the outset of the reform and then again after its second modification. This material guides the central theoretical decisions of the present work toward the elucidation of the Research Questions.

RQ1 How was the programming reform envisioned, deployed and developed?

RQ2 How did mathematics teachers adapt their practices in response to the addition of computer programming to the curriculum?

RQ3 What could constitute a sustained mathematics teaching practice in the presence of programming?

The curricular changes, the teachers as curriculum agents, and the subjects of mathematics and computer programming are the central dimensions that delimit the field of interest. This leads to an overarching approach informed by the work of Michael de Certeau in which tactics and strategies are fundamental terms in the analysis of everyday practices. Consequently, the chapter is organized to expound on the significance of this theory (section 4.1). To finalize, section 4.2 gives a brief overview of another theoretical framework, the
Knowledge Quartet, which was an auxiliary tool for the initial data analysis.

4.1 de Certeau and the theory of everyday practices

Michel de Certeau (1925-1986) was a French cultural theorist with academic roots in history, psychoanalysis, philosophy, and social sciences. His multidisciplinary background is unmistakably present both in his ideas and in his eclectic ways of approaching knowledge. For example, the return of the repressed, with its Freudian and historical reminiscences, is a central idea in de Certeau’s most influential piece, The Practice of Everyday Life. This notion was first developed in his earlier publications, particularly those related to the events of May ’68. These political revolts mark Certeau’s research interests toward contemporary social issues and his later contributions framed in everyday practices.

de Certeau’s theories have impacted a wide range of modern social research, including policy studies in education (e.g. Saltmarsh, 2015; Brewer and Werts, 2017). It is in this framework of contemporary knowledge that de Certeau’s body of theoretical questions, methods, categories, and perspectives are examined to critically analyze teachers’ everyday practices with respect to the curricular and professional constraints that regulate their activities. In the next sections, de Certeau’s theoretical framework is introduced and discussed, highlighting the notions that are considered to be most relevant for understanding how teachers adapt their practices in response to the addition of computer programming to the curriculum.

4.1.1 Everyday practices

In The Practice of Everyday Life (de Certeau, 1984) and its subsequent volume, The art of living and cooking (de Certeau et al., 1998), de Certeau seeks to analyze the humblest concerns of the ordinary people as they are reflected in their everyday practices. He examines and formalizes the means by which the dominant culture is enacted in expectations —social norms, a product’s instructions-of-use, explicit laws— and how these rules are re-appropriated

---

1 The capture of speech and other political writings. de Certeau (1997)
2 First published as l’invention du quotidien, de Certeau (1980)
3 L’Invention du quotidien - Volume 2, Habiter, cuisiner
4.1. THEORY OF EVERYDAY PRACTICES

in everyday situations by the same people upon which expectations are built. In that analysis, de Certeau resorts to the term *culture* in its wider sense as the spectrum of uses and practices of a social group. The core idea that bears his theory is therefore acknowledging that a practice is decisive for the identity of “a dweller or group” (de Certeau et al., 1998, pp. 9)

de Certeau explains that humans make “innumerable and infinitesimal transformations of and within the dominant cultural economy in order to adapt it to their own interests and their own rules” (de Certeau, 1984, pp. xiii–xiv). This process gives intrinsic meaning and agency to the activities of individuals. It creates a private space in which those, otherwise assumed to be passive receivers, establish their way of operating within the arena organized by external techniques of sociocultural production.

Everyday life is thus the life of ordinary people at the micro-level, defined by its inherent messiness and the complex relationship between human actions and ‘the system’. It encapsulates the signs, tools, products, and ideas that are readily available and part of the daily routine. In this system, everyday actions constitute a means of manipulating the common environments in which men and women conduct their lives. Therefore, everyday life also comprises speech, walks, meals, and other expressions of a heterogeneous society in all its diversity.

de Certeau is primarily concerned with describing the emancipatory resistance mechanisms of ‘the ordinary man’ rather than pointing out class struggles and inequalities. The ordinary man is the common hero, the ubiquitous character whose goal is to get by and who does so by *making-do*. Unlike the superheroes of popular fiction, these common heroes are not guided by moral or aesthetic ideals, but by their personal circumstances, needs, and preferences. They are anonymous to the statistics that try to categorize them and to the cultural structures that subdue them but not mere spectators. Instead, they engage in tacit daily practices to pursue their own projects and to evade disciplinary boundaries. Those practices emerge as the unit of analysis in de Certeau’s investigations. Hence, his primary concern lies in the particular schemata of action that appear within a culture disseminated and imposed by the elites. The study of the subjects conducting those actions is for de Certeau, subsidiary to the actual types of operations and it is approached only indirectly.
The theoretical framework defined to examine these infinitely diverse "arts of living" differentiates styles of action according to matter, form, time, place, situations, and circumstances (Rico de Sotelo, 2006). In doing so, de Certeau brings into play the crucial notion of spaces: the space of the other, which one occupies without owning it, the inscriptional space of written words and other semiotic technologies, the interior space, the social space, ... those are for de Certeau the scenes of individual freedom of action, traversed and organized into space narratives. Their opposed polarities are de Certeau’s non-places or nowheres such as the spiritual space, a gap between the everyday and the myth.

It leads to an analysis of the practices organized on three levels: the modalities of action, the formalities of practices, and the types of operations. The modalities of action correspond to the formalities of practices (tactics and strategies, see 4.1.3) and those can be characterized by their different types of operations i.e. the different role of spaces (de Certeau, 1984, pp. 29-30). The same tactical modality could have different meanings and require a different type of operation depending on whether it takes place at home or at work or in this case, in a mathematics classroom (see 4.1.4).

Types of operations, on the other hand, refer to the different strategies and tactics that people use to achieve their goals within a given modality of action. These operations can be deliberate or improvised, and they often involve creative and resourceful solutions to the challenges and constraints posed by the environment. For example, de Certeau describes how pedestrians in a busy city use a variety of tactics, such as walking on the sidewalk or jaywalking, to navigate the urban landscape and assert their presence in public space.

Although they remain dependent upon the possibilities offered by circumstances, these transverse tactics do not obey the law of the place, for they are not defined or identified by it. In this respect, they are not any more localizable than the technocratic (and scriptural) strategies that seek to create places in conformity with abstract models but what distinguishes them at the same time concerns the types of operations and the role of spaces; strategies are able to produce, tabulate, and impose these spaces, when those operations take place, whereas tactics can only use, manipulate, and divert these spaces (ibid., pp. 29, emphases in original)
4.1.2 Power and resistance

de Certeau’s position regarding the workings of power in the spaces of everyday life, from the underside, has its origins in a critique of Foucault’s description of such power relations. For de Certeau, Foucault focused mostly on the imposition of power on the individual and subsequently on the instruments of discipline; the design of environments, and the shaping of mentalities necessary to bring order and to control society. Foucault writes that “where there is power, there is resistance, and yet, or rather consequently, this resistance is never in a position of exteriority in relation to power” (Foucault, 1978, pp. 95-96). This perspective is further explored in the literary dialogue that de Certeau engages with two of the most influential theorists of poststructural and postmodern thought, Foucault and Bourdieu\(^4\) (de Certeau, 1984, Chapter IV).

Power relations, de Certeau argues, go beyond the instrumental view depicted by Foucault, and the structural surrender he sees in Bourdieu’s texts. He suggests that individuals often have some form of agency to either accept these power relations or to resist and manipulate them through the re-appropriation of resources, spaces, language, and narrative. In this dichotomy between power and resistance, Foucault directed his attention toward organizations and the centralization of power. de Certeau expands and responds to Foucault’s views by entrusting agency and autonomy to the governed. The flow of power is for him uneven but nevertheless bidirectional. The intrinsically complex power relations are often reconfigured and subverted in the spaces of everyday actions, in a surreptitious and yet emancipating dialogue with authority.

de Certeau’s focus is not on the large-scale collective revolts but on small, local resistances without expectations to overthrow the elites that represent power. Resistance, as used by de Certeau, refers to actions whose purpose is to constrain change or to inflect small adjustments toward individual goals, not as much an organized political action as an identity resource. It is an act of opposition manifested in verbal, cognitive, or psychical behavior that is not always intentional and not always explicit or externally recognized as such. Importantly, from his perspective, these practices of resistance do not nec-

---

\(^4\)Both Bourdieu and Foucault, together with Habermas, and are critical theorists whose work has extensively influenced critical IS research.
essarily exist outside of dominant institutional frameworks and are possibly most exposed in times of institutional transition (Brewer and Werts, 2017), in this case, a new curriculum.

The latter is essential in understanding how mathematics teachers adapt their practices in response to the addition of computer programming to the curriculum. Their actions are not to be understood as an attempt to reject the integration of programming in mathematics but as a refined and inconspicuous art of teaching according to their own beliefs and within the imposed constraints. The small discrepancies are of private character and do not aspire to transcend to the power spheres. Nevertheless, their actions illustrate how power is fecund and performative, as Foucault astutely remarks:

We must cease once and for all to describe the effects of power in negative terms: it ‘excludes’, it ‘represses’, it ‘censors’, it ‘abstracts’, it ‘masks’, it ‘conceals’. In fact, power produces; it produces reality; it produces domains of objects and rituals of truth. The individual and the knowledge that may be gained of him belong to this production (Foucault 1977, p.194).

The knowledge produced through power dynamics can be obscured by its surreptitious ways, and gathering and analyzing it can help us towards a more informed and nuanced understanding of the unfoldings of the reform.

### 4.1.3 Tactics and strategies

As de Certeau theorized the power of the governed, he located these non-regulated actions of resistance in everyday practices and developed the two central ideas of his theory alongside everydayness: tactics and strategies.

Many everyday practices (talking, reading, moving about, shopping, cooking, etc.) are tactical in character. And so are, more generally, many ‘ways of operating’: victories of the ‘weak’ over the ‘strong’ (whether the strength be that of powerful people or the violence of things or of an imposed order, etc.), clever tricks, knowing how to get away with things, ‘hunter’s cunning’, maneuvers, polymorphic simulations, joyful discoveries, poetic as well as warlike. (de Certeau, 1984, p. xix)
The dichotomous construct between strength and weakness is a fluid notion that evolves and takes different forms along with de Certeau’s ideas. It acknowledges the fact that there are multiple systems of hierarchy and that individuals can be simultaneously powerful and powerless within different systems. de Certeau uses the term ‘weak’ to describe those that, in a specific hierarchy, are forced to operate in a space that they do not control (de Certeau, 1984, p. 23).

According to Certeau, a tactic is “a calculated action determined by the absence of a proper locus [that becomes] the space of the other” (ibid., p. 37). It is characterized by a self-defined target and ways of operating that range from skillful know-how to trickiness, what the Greeks called mêtis. A tactic denotes the creative subversion of the rational order. It re-organizes the products of the given cultural economy for its own purposes, surreptitiously, without taking it over in its entirety.

Playing in foreign territory with alien elements, taking opportunities and looking for a breach, tactics are the art of the weak. They represent creative inconspicuous trickery in opposition to the visibility and regularization that bound strategic moves. It is the lesson of David and Goliath, about understanding the dynamics in a power struggle. Goliath is a giant but cannot pursue the weak, he has heavy armor while David relies on mobility, distance, and chances.

If tactics are the art of the weak, strategies are the methods of the strong, the dominant. Tactical, resistant practices exist alongside the strategic structures of power that characterize institutions. In de Certeau’s words,

> a strategy is the calculation (or manipulation) of power relationships that becomes possible as soon as a subject with will and power (a business, an army, a city, a scientific institution) can be isolated. It postulates a place that can be delimited as its own and serves as the base from which to handle relations with an exteriority of targets or threats (ibid., p. 35, italics in original).

Making political decisions and writing policy texts, what Foucault refers to as discursive practices (Foucault, 2002, p. 51) is therefore here analyzed in terms of strategies, which intention is to drive classroom practices towards the system’s view of the ideal situation.
Which is the place from which curriculum originates and which hazards menace that place? In the entangled institutions of modern society, it might be unclear what constitutes “the country surrounding the city” (de Certeau, 1984, p. 36). It could be as vague as the risk Sweden faces of losing ground as a competitive technological nation or the idea of educated citizens as a pillar of our democracy.

The curriculum is nevertheless a strategic document both here and in a struggle being waged on another less diffuse front; the struggle of the teachers’ everyday practices that is central to this thesis. Actions at the policy level, such as curricular reforms, might be considered strategies in such that they are carefully planned operations from a place of predominance to which teachers do not have direct access and whence punitive measures can be administered.

Considering teachers’ practices in terms of tactics was proposed by Ollin (2005) “to counteract a prevailing ‘culture of victimhood’ whereby teachers are portrayed as powerless (2005, p. 151)”. Instead, Ollin proposes a liberating discourse that legitimates the actions that individual teachers take to retain their autonomy and professional identity. In the present case, focusing on the tactics thus helps to bring out the agential initiatives of the teachers within a centrally governed education system. Teachers are no longer perceived at the bottom of the educational hierarchy, but rather as the essential pillars of the reform, making informed decisions and actively shaping the learning environment.

Further de Certeau explains that the success of institutional strategies depends on their stability and hegemony but that there are always fissures compromising their authority. Thus, those in power engage with methods of monitoring and surveillance but do not necessarily deploy their resources for a lesser cause “The more a power grows, the less it can allow itself to mobilize part of its means in the service of deception (de Certeau, 1984, p. 37)”. The ephemeral fissures in the strategies are the spaces where the tactics flourish. In this sense, strategic positions account for the places and the resources that they govern while tactical moves account for time and opportunities. Tactics comprise therefore an accommodating component. They apply in transient and ambiguous situations and evade precise measurement.
4.1.4 Anti-disciplinary practices

Among the types of operations that could be found among the tactics, de Certeau distinguishes *poaching*, *bricolage* (do-it-yourself), and *la perruque* (the wig). Those tactics have the characteristics of anti-disciplinary practices and workarounds. Unlike other more passive tactical responses, these types of operations presume premeditated choices and a vigilant attitude.

This distinction is related to the way de Certeau equates consumers with the dominated and producers with the dominant. de Certeau suggests that consumers’ ‘ways of operating’ create a ‘network of antdiscipline’ through which the groups and individuals tactically challenge the strategies of the dominant groups, the elite of producers. Consumers can choose to act upon products the way it is intended, or they can invent alternate usages and ‘make-do’ for their individual goals and purposes. de Certeau compares two forms of consuming: watching television and reading. For him, reading appropriates the discourse and is a form of exposing oneself to new, but self-chosen information. The reader is an active consumer not only in the selection of the text but also in the way the text is shaped and understood. The written word takes intellectual effort and interpretation. The man watching television is a passive consumer at the mercy of the producers.

*La perruque* is an expression that refers to the work one does for oneself under the guise of doing work for one’s employer. The worker who indulges in *la perruque* (the wig) actually diverts time which differs from pilfering in that nothing of material value is stolen. It differs from absenteeism in that the worker is officially on the job (du Gay, 1996).

The concept of *la perruque* helps de Certeau to explain how a place of work could be turned into a space of enjoyment, but also how deception and false compliance can manifest themselves as tactics of the weak while only conformity becomes visible in the public. This is also a form of ‘švejkism’ , a “subtle subversion that is invariably ‘invisible’ to his superiors, and often to his peers too” (Fleming and Sewell, 2002, p. 859).

The vocabulary chosen by de Certeau often inherently points to forms of resistance against the authoritarian power of the social order. Alongside la

---

5 de Certeau is writing in the 80s, before the technological revolution
6 In Jaroslav Hašek’s First World War satire, Švejk is the eponymous soldier who finds ways to adapt his duties in the army to achieve his own goals.
perruque, he writes also about *braconner* or *poaching* to refer to the type of operation that involves occupying a space that is not one’s own. *Bricolage* or *do-it-yourself* is a type of operation that is not necessarily clandestine but implies an unorthodox usage of a cultural product. A creative deviation that serves the intentions of the consumer. “Pushed to their ideal limits, these procedures and ruses of consumers compose the network of an antidiscipline” (de Certeau, 1984, p. xiv-xv). Antidisciplinary practices appear as types of operations that convey opposition in more or less concealed forms; from overt resistance and subversion to subtle contention where only conformity becomes visible in the public. Between the “art of placing blows,” and “getting around the rules of a constraining space” (ibid., p. 18), there is a difference in the conspicuousness that is also present in the ways that teachers determine their tactics.

### 4.1.5 Methodological notes

The work of Michael de Certeau can also be read as a meta-methodological argument that insists on our obligation to connect to the real in the face of epistemological skepticism (Highmore, 2007). If there is no ultimate truth to be known about reality, then what would be the point of just offering a version of it? For de Certeau, there is no choice but to work in a world of partial views and embrace situated knowledge. Social studies should encourage heterogeneity and allow alterity to proliferate. de Certeau proposes to introduce what he calls heterology. Rather than seeing plurality and subjectivity as a compromise in the face of the defeat of ‘pure’ objectivity, we should set up some of the conditions for a more profound contact with the real (cf. Critical Realism). The intellectual, in his work towards the object of study, must therefore attend to the implicit, informal, non-verbal data. These insights guided the methodological decisions described in the following chapter, by which data collection is extended to include unit plans and teaching activities with programming from which to access the tacit decisions and the inconspicuous absences.

A second methodological conclusion is to acknowledge the power axis inherent in every research project which de Certeau means is often ignored or suppressed in the literature on methodology in the social sciences. This means to concede that there is no symmetry in the relations that the researcher enters or merely observes and that being aware of the underlying hierarchies will
render a more honest discourse. For this effect, I include subsection 5.7.2 addressing positionality.

Finally, research is for de Certeau a tactical movement itself, in the space of others, taking opportunities and re-appropriating the information to create new results, making-do with the unpredictable.

The ideas exposed here are based on de Certeau’s two volumes on *The Practice of Everyday Life* (de Certeau, 1984; de Certeau et al., 1998) that delve into the private practices and its subtle tactics of resistance that make the art of living. Secondary sources have been found in the narrations of de Certeau’s collaborators Luce Giard (1991) and Carmen Rico de Sotelo (2006).

### 4.2 The Knowledge Quartet

The Knowledge Quartet (KQ) is a theoretical framework for the analysis and development of mathematics teaching (Rowland et. al., 2005). It was initially expected to serve as an analytical tool that would grant insights into the way teachers understand the incorporation of computer programming into mathematics. The framework is summarized here because of its contribution to the identification of the two teaching approaches described in this work. The term Knowledge Quartet was an attempt to operationalize the somehow cumbersome paradigm of Pedagogical Content Knowledge (PCK, Shulman, 1986) that conceptualized “both the link and the distinction between knowing something for oneself and being able to enable others to know it” (Rowland et. al., 2005, p. 255). The knowledge quartet by design simplifies the categories from the PCK model into four dimensions —Foundation, Transformation, Connection, and Contingency—, whose purpose is to structure mathematics teachers’ practice and expertise. Contributory codes exemplify and concertize the most important aspects of each dimension. The four dimensions are summarized in Table 4.1.

*Foundation* includes the knowledge of mathematics itself and knowledge of mathematics education. *Transformation* includes Shulman’s idea of pedagogically powerful forms such as analogies, illustrations, examples, and proofs that make the subject understandable for others. *Connection* presents the link between the internal consistency of mathematical content and its external consistency with other subject matters, together with an understanding that en-
### Table 4.1: The Knowledge Quartet: dimensions and contributory codes as presented in Thwaites et. al. (2010). The codes relevant to the analysis of the data in this thesis are highlighted in bold lettering

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Knowledge and understanding of mathematics per se and of mathematics-specific pedagogy, beliefs concerning the nature of mathematics, the purposes of mathematics education, and the conditions under which students will best learn mathematics.</th>
<th>awareness of purpose adherence to textbook concentration on procedures identifying errors overt display of subject knowledge theoretical underpinning of pedagogy; use of mathematical terminology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformation</td>
<td>The presentation of ideas to learners in the form of analogies, illustrations, examples, explanations and demonstrations</td>
<td>choice of examples choice of representation Use of instructional materials Teacher demonstration (to explain a procedure)</td>
</tr>
<tr>
<td>Connection</td>
<td>The sequencing of material for instruction, and an awareness of the relative cognitive demands of different topics and tasks</td>
<td>anticipation of complexity decisions about sequencing connections between procedures connections between concepts recognizing conceptual appropriateness</td>
</tr>
<tr>
<td>Contingency</td>
<td>The ability to make cogent, reasoned and well-informed responses to unanticipated and unplanned events</td>
<td>deviation from agenda responding to students’ ideas use of opportunities teacher insight during instruction</td>
</tr>
</tbody>
</table>

Enables the teacher to organize and sequence the content. **Contingency** is the name used to encapsulate everything that allows the teacher to successfully deal with the unexpected events surrounding the lesson, what Lindensjö and Lundgren call *realization arena* (see section 2.2). This set of concepts and an associated vocabulary were useful in identifying and describing two distinct tactical approaches by which mathematics teachers direct their choices and actions.

While these dimensions are all related to the teacher’s practice, they are often examined independently to highlight specific aspects of their professional knowledge. In the present investigation, it was the third dimension, connection, that was expected to render fruitful observations to advance our
understanding of a combined programming and mathematics teaching. The subcategories (contributory codes) that characterize this dimension seem to be closely associated with the integration of two fields of knowledge and the joint level of sophistication.

Connection is further explained in the following quote:

[Connection] binds together certain choices and decisions that are made for the more or less discrete parts of mathematical content —the learning, perhaps, of a concept or procedure. It concerns the coherence of the planning or teaching displayed across an episode, lesson, or series of lessons. (Rowland, 2013, p. 24)

The sequencing of topics is a part of the coherence and it can be observed in planned instruction within and between lessons. It includes the ordering of tasks and exercises that can be inferred from deliberate choices and awareness of the relative cognitive demands of different topics.

The theories and frameworks exposed in this chapter present coherent guidelines to analyze and explain the teachers’ practices revealed in the data. de Certeau’ provides terms and mechanisms to describe both modalities of action, that is, the strategies in the policy documentation and the correspondent tactics with which teachers respond. The activities that the mathematics teachers propose to include computer programming suggest tactical maneuvers in line with several of the types of operations that de Certeau devised in everyday practices.

### 4.3 The practice of teaching

The practice of teaching is a central notion in the analysis that shall be explored in this thesis. Practice in this context is seen as a socially organized activity recurrently undertaken,—what the practitioners do, rather than who they are or how they think. Studying practices allows researchers to engage in a direct dialogue with practitioners and opens for examination issues that are directly relevant to those who are dealing with strategy.

The idea of practice is ever-present in the work of Michel de Certeau and also in Rowland’s Knowledge Quartet. de Certeau considers practice as “a
way of thinking invested in a way of acting, an art of combination which cannot be dissociated from an art of using.” (de Certeau 1988 p. xv) adding the inventiveness and ‘Do it yourself!’ techniques to the endeavor. ‘Thinking’ does not leave out less deliberate forms of practice. Rather, de Certeau acknowledges also oblivious practice:

The ordinary practitioners of the city live “down below”, below the thresholds at which visibility begins. They walk [] follow the thicks and thins of an urban “text” they write without being able to read it. These practitioners make use of spaces that cannot be seen; their knowledge of them is as blind as that of lovers in each other’s arms. It is as though the practices organizing a bustling city were characterized by their blindness. (de Certeau, 1984, p. 93).

Saltmarsh (2015) takes a step further and adds that it is the strategies produced by the dominant system, together with the tactics that create sustained practices.

With the practice of teachers in mind Schatzki (2000) reframes the idea of awareness, where also blind ‘uses of space’ bear meaning in the teachers’ professional practice. He exemplifies how a nondeliberate teaching activity is guided and structured by goals that define its nature and purpose.

These “in order tos”, “toward whichs”, and “for the sake ofs” —in more conventional terms, purposes, tasks, and ends— orient [the teacher’s] activity in the sense of structuring what he is up to: in conjunction with the current situation, they specify that writing on the board makes sense —that is, is the appropriate and needed thing (given the situation and the purposes and ends involved). They determine, in other words, what might be called the practical intelligibility that informs non-deliberate activity. (ibid., p. 33, italics in original)

The starting point could be chosen to be Phronesis, the practical wisdom to act in the moment on the basis of general principles (Aristotle). That leaves two entry points for policy to influence practice: the authoritative and the educational.

Policy can pragmatically constrain the options that are available to act on at a given moment but it can also patiently induce new general principles that
4.3. THE PRACTICE OF TEACHING

resonate with teachers’ understanding of their own practice and can be integrated into their contextual decision-making. This point of view with regard to teachers’ practice and agency is further developed in the work of Giroux (2018). Giroux defends the importance of the intellectual teacher, who is able and willing to develop and critically appropriate curricula. Teachers are not seen as mere implementers of curriculum and their practice is therefore the result of a deliberate choice, even if not all the decisions are equally conscious. Giroux reflective practice postulates a teacher capable of examining how schooling in general, and their own teaching specifically, contribute to the goals of education.

In relation to the tactics, there are several competing notions that aim at capturing the diversified and often conflicting goals that teachers navigate in their practice. Skott (2001) talks about critical incidents of practice with mathematics teaching in focus. Here, the teachers’ agency becomes part of the practice when they conciliate the curriculum in interaction with their students. i.e., it places teaching within its situational and institutional context.

The practice of teaching refers not only to the various teaching actions involved but also to the meanings that teachers assign to what they do, and their motives. In the realm of mathematics teaching, Rowland (2009) uses practice in two different senses. When talking about the mathematics teachers’ practice, it reflects expertise in their profession, a habitual way of operating. From the learner’s perspective, on the other hand, practice is seen as the act of doing in order to familiarize oneself and improve, often by means of repeated exercises. It is the mathematics teachers’ practice that is considered here in ‘the practice of teaching,’ with the ambition of discussing the habits that appear along policy changes. Regardless of whether they are deliberate or oblivious, it is the recurring aspect of the actions that make them become practice.
Chapter 5

The Empirical Study of Programming in Mathematics Education

This chapter is devoted to the description and justification of the selected research methods for collecting and analyzing data. The study is designed to add to the descriptive knowledge of teacher professional autonomy and ultimately, the didactic decisions that build their sphere of influence. Teachers themselves are therefore chosen to provide that knowledge, and the opportunity to attend to their own voices and their own stories would be the primary way to secure the message.

In the following pages, I present a chronological portrayal of the processes that led to an explanatory model addressing the research questions. Chronology is essential in this thesis because the series of events surrounding the implementation of the reform had a significant impact on the questions, but also on the method, and ultimately, on the findings and conclusions of the study. In other words, the thesis would have been very different had the study been conducted at a different time. These particular social and historical processes shaped the investigation, from the incipient days of a novel reform to re-prioritization amidst a global medical emergency and the mature reflections of the recent years (see Figure 5.1). This underscores the importance of
conducting context-sensitive research that acknowledges and accounts for the ever-changing nature of social phenomena.

The chapter begins with a description of the research design in connection to the research questions, including the qualitative research paradigm and other necessary methodological adjustments adopted during the investigation (section 5.1). The two main sources of data and their respective analysis are the topics of sections 5.2 and 5.3. The chapter concludes with ethical considerations and personal reflections on methodological development and trustworthiness (section 5.7).

5.1 Research Design

The purpose of this thesis is to understand how mathematics and programming converge in teaching activities and the reasons behind the decisions that teachers make within and at the boundaries of curricular constraints. Gradually, both the surrounding constraints and the reasons evolve, raising new practices. Making sense of these phenomena is important not only to inform future curricular reforms but also to open up new possibilities in current implementation efforts. With this in mind, the study was designed to uncover the curricular constraints and delve into the teachers’ practices as they change over time. The research presented here aims at describing and explaining, —rather than predicting or controlling— going beyond the particular teacher narrative and trying to connect it with similar descriptions and events. This section will therefore describe the overall approach grounded in critical realism and the qualitative methods used for data collection, analysis, and interpretation of results.

5.1.1 Methodological frame of reference

Methodology involves the set of assumptions and procedures linked to the philosophical intent of the research questions and provides the underlying logic and rationale for the research process. Therefore, the philosophical orientation has implications for the choice of method and needs to be acknowledged.

This thesis is anchored in the fields of Information Systems (IS) and Work
Integrated Learning (WIL). The qualitative studies in these disciplines have long been influenced by hermeneutics, interpretivism, and phenomenology (eg. Boland, 1985; Butler, 1998; Walsham, 2006; Stephenson et al., 2018). Nevertheless, the depths of teachers’ practices seem to be better understood once the lenses of these paradigms are augmented with a Critical Realist perspective (CR). This meant abandoning the belief that social reality is solely constructed through the interpretations of individuals. Critical Realism introduces a layer of the real, mechanisms that exist independently of the observable phenomena and our interpretations of them. Bhaskar (2010) corroborates this shift with the disclaimer that CR does not aim to substitute other conceptualizations of reality but instead to add a necessary perspective able to explain “structures generating social phenomena” (ibid., p. 3). This stratified ontology of events, mechanisms, and experiences has proved generative in understanding power and the influence of policy in the educational sphere (eg. Ball, 2003; Perryman et al., 2017). This investigation embraces what Ball (2003) calls the double-edged connotations of CR as the instrument for linking the experiences of teachers to the encompassing policy organizations,

the policy technologies of education reform are not simply vehicles for the technical and structural change of organizations but are also mechanisms for reforming teachers […] and for changing what it means to be a teacher, the technologies of reform produce new kinds of teacher subjects (ibid., p. 217).

Critical realism seems therefore well suited to bring forward the meaning of experience for the constitution of social reality and the centrality of action and practice within structures of power advocated by theorists of the ‘practice turn’ such as Michel de Certeau.

5.1.2 Research journey

The empirical data that substantiates the argument of teachers’ practice being linked to institutional structures were collected at two separate points in time, hereafter called the first and second iterations. An outline of the research journey is depicted in Figure 5.1.

For the purpose of the study, a combination of document analysis and semi-structured interviews were the primary data collection methods. During
the first iteration, various documents related to the new mathematics curriculum were analyzed, followed by interviews with teachers. During the second iteration, interviews and document analysis continued to be the primary data collection methods. The follow-up interviews provided a deeper understanding of the changes and developments that occurred since the first iteration. I specifically conducted follow-up interviews with the same group of teachers to track changes in their experiences and perceptions over time. I also analyzed new documents, such as updated curriculum materials and teacher reflections, to capture relevant shifts in instructional practices.

5.2 Curricular documents and strategic positions

The departure point for the data collection was gathering the documents related to the introduction of programming in the mathematics curriculum. de Certeau suggests that

A practice of the order constructed by others [a network of already established forces and representations] redistributes its space; it creates at least a certain play in that order, a space for maneuvers of unequal forces and for utopian points of reference (de Certeau, 1984, p. 18).
That is, where there is a dominant social system, there is always a preferred interpretation, a particular way that system is designed to be used. This section describes the methods that are used to disclose that “certain play” in the educational order.

The contemporary organization of society as well as the narratives constituting the public sphere are mediated by texts. More recently even video productions, particularly in the context of online education can be considered a part of the official communication, the *inscriptional space* (p. 36). The message, textually or otherwise mediated and disseminated has a distinctive character and capacity, that of producing the same set of words in multiple sites, simultaneously or asynchronously. The reproduction and continuity afforded by texts are essential for large-scale administered organizations such as curriculum implementations. Texts ratify the stability of policy allowing it to reach schools separated from one another not only in space and time but more importantly across the social context.

To assess the direction of the curricular reform, the regulatory documents available at the time of the investigation were scrupulously analyzed. These documents include the Educational Act, the curriculum of 2011 (Gy11), and the following amendments to the curriculum itself. Furthermore, the analysis comprises the comments to the curriculum aiming at clarifying and exemplifying the new directives and the material from the official teacher training workshops that the Agency for Education offered both online and on-site. Table 5.1 provides a comprehensive summary of the documents included in the analysis.

The governmental bills regarding the introduction of computer programming provide further insights into the motives and expectations of the reform as well as the resources deployed in that direction (eg. Regeringskansliet, 2018). In one of the initiatives supported by those assets, Swedish universities were invited to submit a tender to provide teacher training courses in text programming. The specifications and suggested syllabus that accompanied the call are also analyzed in the context of their significance for an endorsed content and progression path. This larger corpus is listed in Table C.1.
CHAPTER 5. EMPIRICAL STUDY

Table 5.1: Summary of data sources for the strategic analysis

<table>
<thead>
<tr>
<th>legal binding &amp; visibility</th>
<th>source and Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laws, bills and directives</td>
<td>Educational Act (Skollagen, 2010)</td>
</tr>
<tr>
<td></td>
<td>Regeringskansliet (2017c), Regeringskansliet (2017a), Skolverket (2019a), Regeringskansliet (2017a), Regeringskansliet (2018), Skolverket (2022a), Skolverket (2022c), Regeringskansliet (2023)</td>
</tr>
<tr>
<td>Teaching Goals</td>
<td>Curricula (Skolverket, 2019b), Skolverket (2017c), Skolverket (ibid.), Skolverket (2016), Skolverket (2019a), Skolverket (2022e)</td>
</tr>
<tr>
<td>Clarifications and examples</td>
<td>Skolverket (2017a), Skolverket (2019c), Skolverket (2022d)</td>
</tr>
<tr>
<td></td>
<td>IT-strategies (Utbildningsdepartementet, 2015; Utbildningsdepartementet, 2017)</td>
</tr>
<tr>
<td>Teacher professional development</td>
<td>Statensskolverk (2018) and Statensskolverk and Grapenfelt (2019)</td>
</tr>
<tr>
<td>Invitations to tender</td>
<td>25 official course specifications (listed in Table C.1)</td>
</tr>
</tbody>
</table>

Not less significant for the outcome of the reform are the absences and gaps in the documents that constitute the strategies. This means that not only documents explicitly addressing programming need to be analyzed, but also those in which new curricula are often reflected. This information was retrieved during the second iteration of the project to corroborate the presumed absences that were made evident during data analysis in reference to the National Exams.

In this respect, two kinds of data were considered necessary and relevant to examine further strategic resolutions. First, the publicly available documents surrounding the National Exams in mathematics were analyzed. These included exam examples, teacher and pupil instructions, and ordinances regulating current and future exams.

To complete and contrast this data, interviews were conducted in person and via email, with staff members from the two independent institutions responsible for developing the tests: the PRIM group and BVM. The PRIM

---

1Exams and tests are henceforth used as synonyms without distinctions between the two concepts. National Test is the preferred terminology by the National Agency for Education. Scholars, however, refer to tests as tools to measure the knowledge level of students in order to adjust the learning material accordingly whereas exams or examinations are more formal and summative.
group\textsuperscript{2} is a research and test development body with authority on the assessment of knowledge in mathematics. They are responsible for the National Exams concluding the compulsory levels in ISCED 1 and ISCED 2 and the first year of non-compulsory education (ISCED 3) which is common to all the programs. This gives their measurements the advantage of continuity and an understanding of the progression within the subject across levels.

BVM\textsuperscript{3} is a research group in the Department of Applied Educational Science Research at Umeå University. They develop the National Exams in mathematics for the rest of the ISCED 3 levels and conduct quantitative studies on grading in the classroom, test development, and selection to higher education. The interviewees from these two organizations provided further information from complementing assessment perspectives. They were asked about the constraints and possibilities that the new curriculum entailed as well as insights into the process of developing National Exams for the different ISCED 3 levels. The answers from these two test developers are referred to as TD1 and TD2.

A comprehensive description of the intricacies of research of institutional texts by Miller and Dingwall (1997) reveals how this pervasive amount of files, records, and statistical reports is well suited for qualitative analysis to advance the understanding of the contemporary institutions to which they are inherently attached. Through reconstruction and reexamination, but avoiding over-analyzing and pursuing practical considerations, these texts can help researchers to challenge the views of social reality and to enrich observational studies (ibid., p. 87).

With this intention in mind, the analysis of the official documentation was developed from two different readings with different lenses. The first reading rendered a general overview of the information and positioned the study in the historical and legal frame generously extended by the documented sociocultural praxis (chapter 2). The second reading was thoughtfully directed to find evidence of the concrete strategic choices that could have the most impact in shaping teachers’ tactics, particularly in relation to the new results pointing at teachers’ divergent views of the pedagogical progression embedded in the subject(s).

\textsuperscript{2}PRIM https://www.su.se/primgruppen/
\textsuperscript{3}BVM Measurement in Behavioral Sciences (Swedish: Beteendevetenskapliga mätningar)
CHAPTER 5. EMPIRICAL STUDY

These policy records were reviewed in search of explicit clues in relation to what in de Certeau’s terms could be regarded as the officially sanctioned strategies (see subsection 4.1.3). Passages mentioning programming but also omissions in doing so revealed the intentions and exertions in the cause and framed the level of autonomy in teachers’ professional practice.

5.3 The voice of the teachers

This section describes how the empirical data needed to address the research questions were generated and analyzed. A valuable source in retrieving information about teachers’ integration practices is the teachers affected by the changes and whose responsibility is the pedagogical design that will guide classroom instruction. The abstract nature of inquiry about motives and autonomy delves into context-specific questions about how the addition of computer programming might have shaped mathematics teachers’ practice and to which extent this reshaped practice conveys the aforethought purpose of the reform. These issues were addressed by examining nine concrete examples in which teachers shared their decisions and insights in relation to their classroom instruction.

The semi-structured interviews with teachers allowed the researcher to gain an understanding of teachers’ autonomous choices from their own perspective and provided depth of data (Patton, 2002). Complementing data was retrieved from teachers’ unit plans, lesson materials, and to some extent, on-site classroom observations.

For the purpose of investigating which variables might have influenced an extended practice in teaching mathematics with programming, the participation of the teachers was granted at two separate points. The researcher arranged follow-up interviews with the teachers who, after a time-lapse of two years, still continued using computer programming in their mathematics classrooms.

4Unit plans comprise several lessons and are made to serve for a long period of time during which the topics in the unit are studied. Figure 5.2 shows an example of the information that might be included in such documents.
5.3. EMPIRICAL STUDIES

5.3.1 Selection of participants

The investigation started the first year after the new curriculum began to apply. To concentrate on the activities resulting from the actual implementation, it was necessary to find mathematics teachers in Upper Secondary Schools who already knew computer programming. This condition effectively sidestepped issues of teachers learning programming and teaching it at the same time, which would have added another transitory dimension. At this point, and due to the rapid implementation of the reform, the targeted population was scarce (Larsson, 2017; Misfeldt et al., 2019). For this reason, it was considered necessary to recur to a purposive selection of participants that could provide realistic accounts of how programming was being integrated into their mathematics lessons. This method belongs to the category of non-probability sampling techniques which means that the participants are chosen on the basis of their knowledge, relationships, and expertise regarding the phenomenon under investigation (Palys, 2008). In the current study, the nine participants are all experienced mathematics teachers proficient in computer programming (see Appendix 12.3).

The opportunities to contact mathematics teachers who were actively implementing the new curriculum had three fronts. The majority of participants were recruited through the initiatives that the Agency for Education had already put in place in 2017, including their training programs and programming workshops for teachers. These initiatives had enrolled as teaching assistants a group of mathematics teachers that could support their colleagues in learning programming. Professional online forums for mathematics teachers provided another useful contact network to find teachers who were already working with programming. Finally, the Klein-Days teacher training arranged by the Mittag-Leffler Institute provided an additional platform to connect with teachers who were open to using programming as a tool for teaching mathematics.

From these partially overlapping cohorts, 15 mathematics teachers active in ISCED 3 were reached via email. Nine of the teachers were finally chosen for the first round of interviews. Their willingness to participate together with an ongoing teaching practice in which programming and mathematics converged were critical elements in selecting these teachers.
The nine participants that provided the empirical data came from eight different upper secondary schools across Sweden and taught different mathematics courses along with some other subject(s). They were all licensed to teach mathematics at the upper secondary level and had relevant programming experience (see Appendix 12.3). The respondents taught in the technical, natural sciences or economics programmes where students have the possibility to complete up to 6 consecutive mathematics courses during three years after compulsory school (ages 16–18). Although there was not an even distribution regarding which courses were taught or which programs were represented, there was no intent to draw general conclusions in this sense, rather, each teacher’s testimony was individually analyzed and each stood independently with equal importance in the data analysis. These dramatis personae are not to be considered as representatives of different teacher typologies, but examples of a much broader professional spectrum. Nevertheless, their common concerns but also their differences can guide the data towards highlighting different aspects of both personal and collective visions of teaching mathematics with programming.

Cooperation with the researcher was granted based upon the assurance of confidentiality. Participants were informed of their rights in a letter accompanying the invitation. Their right to interrupt or withdraw participation at any moment was made explicit, as well as the possibility to access the ongoing documentation and retract from all or any part of it. To protect participants’ identity, and in compliance with GDPR, the actual districts, school names, and locations were concealed. Care was also taken to disguise the gender identity of the teachers and other contributors by referring to them with gender-neutral aliases and the singular they\(^5\) pronouns (Bjorkman, 2017; APA, 2020). The names of participants mentioned in this study are fictitious and any similarities to actual persons are coincidental and unintended.

### 5.3.2 Interview design and implementation

The data for the investigation were collected at two different points in time. The first iteration of the study was conducted soon after the revised curricu-
5.3. EMPIRICAL STUDIES

lum was deployed, between September 2019 and April 2020. At that time, a total of nine teachers were interviewed. These initial interviews were conducted mostly at the schools where the teachers were employed, often in connection with classroom visits. Two of the interviews took place online. For the second iteration of the study, conducted in the spring semester of 2022, I managed to trace back and re-contact all nine teachers in the original sample. Seven teachers from the initial cohort were still engaged in mathematics courses with some degree of programming and they were willing to resume the interviews. Taking into account the traditionally high turnover among teachers, and that the scope of programming had been diminished in the second revision of the curriculum, this attrition can be considered within reasonable bounds. During the second iteration, online interviews were by far the most common choice, using varying video conferencing software depending on the teachers’ preferences (see Figure 5.2). This gave also a peek into their recent experience with distance teaching methods.

This kind of research design conforms with Epstein’s third category for qualitative longitudinal studies in which the researcher returns after a lengthy interval of time has elapsed (Epstein et al., 2002, p. 64). Nevertheless, it must be acknowledged that, because the Covid-19 pandemic abruptly interrupted the expected course of many educational projects during those years (Jandrić, Fuentes Martinez, et al., 2022), the acclimatization phase that Epstein anticipates might have been severed.

There are many ways of studying teachers’ practice to inform different aspects of the multifaceted teaching context. (Hill et al., 2007) document a long history of assessing mathematics teachers’ practices with methods ranging from written tests, lesson observations, and self-reports. They also allude to task-directed interviews as an access point into the thought processes and motives in which teachers ground their practice. In this kind of interview, teachers are asked to reason about a concrete mathematical task. The questions revolve around the possible methods to solve the task, to which extent they think that their students will benefit from it, and which type of mathematical difficulties or misconceptions are likely to require their attention.

A variation of the task-directed interview was used in this study, where the mathematical tasks were replaced by the teachers’ unit plans and related documents. These materials were a visual tool to guide the discussion.
The teachers were asked to bring the unit plans they normally share with the students, together with lesson plans, worksheets, code examples, etc that they considered relevant to illustrate how they included computer programming during the mathematics courses they were teaching. The principal use of these documents was to expand the information available to the interviewer by referring to concrete examples. A particular programming activity could more easily elicit follow-up questions such as how they selected the materials for instruction or how they explained certain concepts and procedures. As predicted by Stephenson et al. (2018), when the researcher is also a practitioner, the interviews are likely to turn into lengthy conversations often interrupted by the teachers’ other commitments rather than the exhaustion of the topics. These fruitful conversations not only provided key information about
5.3. EMPIRICAL STUDIES

the learning goals that directed the chosen teaching activities but opened also unexpected and exciting avenues for further analysis.

To this effect, the interviews followed a semi-structured protocol with three focus areas: a) the role of programming in mathematics, b) examples of programming activities in their mathematics courses, and c) reflections and lessons learned from the conducted programming activities. The interviews started by discussing open-ended questions about teachers’ experiences teaching programming, their motivations for using programming, and the challenges they faced. Later the questions were steered towards the actual programming tasks within the classroom activities that the teachers planned for their students. During the task-directed part of the interviews, the researcher allowed the participants to set the parameters of the topic area and describe their ideas, without assuming their motives. Giving the teachers space to lead the interview might uncover novel and unanticipated teaching experiences.

There were nevertheless, three concrete inquiries devoted to leading the informants into a deeper analysis of their examples. Teachers were asked how they decided *what to teach and how to teach it* in order to relate the choice of activities to the curriculum and instruction. A second question—*What level of activities do you seek?*—invited to reflect on complexity and expected progression sequence. Adjacent demographic data were also gathered, regarding teaching experience, school type, and the mathematics courses in which the participants were involved (Table A.1).

In the second iteration, additional questions were introduced to account for the dynamic aspects of the investigation. These included survey-like questions about the amount of time that the teachers estimated was dedicated to programming activities in their classrooms but also the perceived difficulty of those activities compared to their teaching practices at the beginning of the reform. The idea of continuity was introduced in questions highlighting changes in teaching with respect to the initial interviews—*What do you do differently? What have you kept? Why?*. More importantly, the interviewer encouraged the discussion by resurrecting relevant passages and documents from the first interview. The teachers were also able to look back at the unit plans and classroom activities they had presented during the first iteration. This elicited interesting reflections, providing valuable insights into their own pedagogical growth and the evolution of their instructional approaches. The analysis of
CHAPTER 5. EMPIRICAL STUDY

this data allowed for drawing conclusions about the evolving nature of programming practices in the context of teaching mathematics in Swedish secondary schools.

The interviews were audio-recorded and transcribed for analysis. The participants were briefed and replied with their consent ahead of each individual interview. The length of the interviews varied largely from 20 to 90 minutes, depending mostly on the amount of material that the teacher had brought along and the time their busy schedules allowed. The interviews in the second iteration were on average shorter (20 to 45 minutes). This was partly due to a narrower interview guide but also to the teachers being generally less inclined toward discussing the politics behind the reform, and to a certain extent due to a decreased amount of programming content to be discussed. In total, more than sixteen hours of audio-records were transcribed verbatim.

Written, asynchronous, email conversations were included in those cases in which clarifications or additional materials were rendered necessary after the interview.

It should be pointed out that a precise linguistic analysis of the interviews is outside the scope of this thesis. Therefore, it was considered that the content of the passages should outweigh showcasing the details about the thought processes necessary to articulate the ideas (Poland, 1995, p. 292). For that reason, and to increase readability, pauses, false starts, self-corrections, and some fillers have been omitted when the quotations were translated into English. Other nonverbal elements of the interview, such as pointing at an exercise in the worksheet or running a program on the computer, were documented by the interviewer either in handwritten notes on the interview guide or by saying it aloud to have it on record. In the following passage, the interviewer completed some missing reference information (in bold):

(teacher) — This is more complicated, you see, because there are two, so that is the next step..

(interviewer) — There are two parameters in the rectangle area function...

(teacher) — Yes! so it is actually more like a formula than a function, but we call it a function because this is programming.
5.3. EMPIRICAL STUDIES

This passage would be later condensed to feature only the teacher, which resulted in a single statement conveying the message.

*This [the rectangle area function] is more complicated, you see, because there are two [parameters], so that is the next step […] so it is actually more like a formula than a function, but we call it a function because this is programming.*

For the analysis, the passage would be split into two related ideas according to the unit of analysis. By saying that something “is more complicated”, and mentioning “the next step”, Ariel provides an insight into the progression and the level of cognitive complexity of the task. The second idea in the episode confirms this division. It comes from the statement about nomenclature: “It is actually more like a formula than a function, but we call it a function because this is programming”. Here, Ariel makes a purposeful distinction between the two areas of knowledge, by using the terminology from the programming domain to the detriment of the more nuanced mathematical term.

5.3.3 Unit plans and other teaching materials

There is a reason to believe that some of the tacit tactics that are concealed in the spoken interviews could be found in the ways of operating revealed in other non-spoken information. Following the methodological coordinates offered by de Certeau it is understood that

[T]actics in discourse can, as we have seen, be the formal indicator of tactics that have no discourse. Moreover, the ways of thinking embedded in ways of operating constitute a strange—and massive—case of the relations between practices and theories (de Certeau, 1984, p. 45).

Therefore, teaching materials, particularly those developed by the teachers themselves, can be vehicles of information, unveiling tactics that are not openly recognized in the spoken discourse.

The materials that the teachers brought to the interview served a second more pragmatic purpose: focusing the discussion on the activities that constituted an essential element in teachers’ pedagogical practices. An impor-
tant fount of information were the teachers’ unit plans for the mathematics courses they were teaching. A unit plan is a document that unifies several single lessons into a coherent learning experience. The teachers design their learning program around a particular set of topics over multiple class sessions, lasting normally several weeks. The level of detail of a unit plan might refer to textbook chapters, recommended exercises, or field trip activities and it is usually concluded with some form of assessment. Its purpose is to structure the course and to provide a plan of action by dividing the learning into time-specific blocks. An example can be seen in Figure 5.2, where the teacher shares their plan for the second part of the spring semester.

Along with the unit plans, the teachers provided a variety of course materials that they had used or planned to use in their lessons. These documents included worksheets to be handed out to the students, code snippets used during a lesson and even notes the teacher made after a learning activity to follow up on it. Many of the activities that the teachers were willing to share were self-designed programming exercises and demonstrations that they had added to the course after programming was integrated into the mathematics curriculum. Explaining and discussing these activities took a large part of the interviews, and was not only informative and fruitful but gave each conversation its unique color.

Another collateral merit of the documents was made apparent after the transcript corpus had been analyzed. There it was possible to find further evidence of the tentative division of teaching approaches that was being disclosed. For example, a distinctive characteristic of teachers adopting Dual Teaching tactics was the relation between the general school calendar and the dates on which mathematics lessons were dedicated to computer programming. That relation could first be identified once several unit plans from different teachers were gathered and examined together.

The inclusion of unit plans and course materials in the research data provided valuable insights into teachers’ programming activities, revealing evidence of different teaching approaches, and enhancing the validity and contextual understanding of the interviews. These documents contribute to a deeper understanding of the planned lessons, their relationship to the overall curriculum, and the unique perspectives and practices of the teachers involved.
A third benefit of including the unit plans and worksheets in the data set was that they contributed to triangulating the information and added to the validity of the interviews. Triangulation is possible when different sources of data are aggregated to test the quality of the information, to understand more deeply how the different elements are linked together, and ultimately to put the whole situation into perspective. Having the unit plan as a mediating element of the discussion helped the teachers to talk about what actually was planned for the lessons rather than what they wished they had planned instead or what they thought the researcher was expecting to hear. It also added information about lesson length, time of the day, day of the week, etc, and more importantly, it positioned the activities in the broader context of the succession of topics that characterize the syllabi. A second triangulation of data occurred during the follow-up interviews, in the second iteration of the data collection. At that point, the teachers were encouraged to revisit their unit plans and classroom activities, which I had brought along from the previous interview two years before. This enabled the teachers to correct and clarify possible misrepresentations from the first iteration.

5.3.4 Classroom observations

On four occasions it was possible to arrange for a relevant lesson observation alongside the teaching interview. I was introduced to the class as competent in both computer programming and mathematics and willing to help the students with the lesson’s exercises. Due to this participatory role, I was able to interact directly with the students, and the general impressions about the teaching activity were documented afterward. The approach granted access to information on how the cognitive level of the tasks matched the students’ knowledge and whether their difficulties stemmed from coding or math gaps. On-site observations gave also coherence and depth to the information retrieved in the interviews and were suitable to compare and test the quality of other evidence. For example, despite being stated in the unit plan, the assigned homework for the programming activity was not understood as such by the students.

However, this kind of data was vague and not as fruitful as the teaching materials. Classroom contexts proved to be extremely complex and unforeseeable and the social structures that governed the situation included much more
than the planned activity. The teacher’s attention flickered between dried-out whiteboard markers, charger cables that needed to be plugged in, and a stubborn wasp hitting the windows. As a consequence, the actual pedagogical design that was of interest became blurred off in the realization phase (Lindensjö and Lundgren, 2000).

Another problem with the observations was that they were clearly inclined toward lessons held by teachers who knew well in advance when programming would be on schedule. This bias prevented access to more scattered or informal programming activities that might only take a couple of minutes of the mathematics lesson and which the teacher did not consider important enough to be worth organizing a visit. Furthermore, there is already evidence for existing congruence between teachers’ instructional choices in their planned activities and their subsequent classroom practices (eg. McMullen et al., 2005). For this reason, the triangulation advantages of yet another data corpus on the integration of programming in mathematics did not outweigh the practical problems and the initiative was discontinued.

5.4 Analysis of data

The data from this research was analyzed using the principles of qualitative data analysis. There were a total of seventeen interviews: nine during the first iteration, seven follow-ups, and one with one of the developers of the National Exams (the other was a written conversation). All interview transcriptions were examined and participants’ statements were summarized by identifying the contiguous portions of the text expressing a single coherent idea (Marks, 1990). The next step was to find those passages bearing information about instructional judgments. These ideas appeared in the form of examples, explicit motivations, and expressions signaling choice or lack thereof.

Although software packages like NVivo can assist with data management and analysis, the discovery of themes remains the responsibility of the researcher. In my case, I made the decision not to utilize software for analysis due to the time-consuming nature of mastering and using it. Furthermore, there is a risk of becoming too fixated on predetermined themes as a result (Walsham, 2006, p. 325). Instead, I opted to rely on spreadsheet packages to support my analysis and I organized my case materials using files in folders.
This approach proved effective for the volume of data I had gathered.

5.5 Abandoning the Knowledge Quartet

This section describes the process of analyzing the data and its development. The analysis was guided primarily by the learning activities selected by the teachers themselves to convey mathematics and programming and the motives that supported their choices. Initially, it was expected to produce some kind of common pedagogical content framework for the integration of computer programming in mathematics. However, after a failed attempt at arranging the described activities according to their level of complexity, it became apparent that the interview data had some other story to tell.

The analysis of the transcripts started chronologically and advanced in parallel with new interviews taking place and adding to the corpus. The initial categories were adapted from the Knowledge Quartet: Foundation, Transformation, Connection, and Contingency (section 4.2).

The third dimension regarding the connection and coherence of the planning was selected for a first analysis of the activities and episodes of teaching mathematics with programming. Two subcategories in this dimension—decisions about sequencing and anticipation of complexity—were of particular interest to inform a possible progression for the integrated programming and mathematics curriculum. Also, the codes making connections between procedures and making connections between concepts from the same category seemed to be closely related to the inclusion of programming in mathematics. Identifying the passages akin to these descriptions was expected to organize the data and present a picture of the teachers’ practices in relation to the new blended subject. To this effect, 31 episodes displaying planning or teaching programming across a lesson or series of lessons were located in the corpus during the first iteration, and tentatively pooled with respect to the contributory codes of the category of connection.

However, in the passages in which the programming activity was entangled with mathematics, the classification tended to depend solely on the mathematical content or the programming content. For example, Ariel indicates that is more complicated, you see, because there are two [parameters], so that is the next step [...] [A:1]. While this will qualify as ‘connection’ in the Knowledge
Quartet, the progression refers exclusively to programming skills and not to the merged subject.

For this reason, the framework did not contribute to advancing the investigation in the direction of a model for integrated programming and mathematics. More interestingly, the division of classification grounds alluded to two different ways of including programming. This idea was pursued within the scheme of tactics and strategies proposed by de Certeau (1984) and a grounded approach to data analysis led to the identification of two ways of understanding the integration of programming in mathematics teaching: Dual Teaching and Interspersed Programming (sections 7.2 and 7.3).

de Certeau’s notion of self-defined goals that upheld tactical moves added a new frame of reference to interpret the ways in which teachers were selecting and organizing programming activities. Teachers’ everyday practices were understood as tactics deployed in the space of the other. The lesson content was chosen tactically amidst new curricular directives. The official documents became therefore an operational counterpart to articulate the strategies.

5.6 Mapping and finding tactics

The tactics that de Certeau mentions (subsection 4.1.3) can be seen as a repertoire of actions and possibilities that describe a landscape of both conformity and resistance. These tactics are not necessarily all-encompassing or exhaustive, nor are they independent or disjoint from one another. Such would be the case when Francis reuses a linear optimization problem from the textbook, in which the goal is to determine the number of units of each product in order to maximize the total profit (F:1).

In the following testimony, Francis recognizes that the activity, despite using programming, does not require the students to engage with code. By doing so the teacher hoodwinks the students into complying with the curriculum without friction. In this sense, the activity can be seen as disguising an intention by assuming the metaphorical perruque. At the same time, they build alternative models from the existing praxis as creative bricolage tactics made from the pieces at hand, for their own purposes. Both tactics la perruque and bricolage coexist and explain the activity, and are seen as such in the foreground of the strategies.
5.7. TRUST AND CUMULATIVE KNOWLEDGE

There are libraries like JOptimizer or Colt to solve linear optimization, but that’s not the issue, because the struggle here is to set the equations right. 2 hours of labor and 3 units of raw materials for the first container, whereas it takes 4 hours of labor and 1 unit for the thin one. How do we express that? How do we include the constraints on labor and the limits on raw materials? But before that, they need to understand that different combinations give different profits and that some combinations are not possible because of the limitations. So I made the program that does just that [show the result of the different combinations], and we tested it. A bit like a game, to see who came closest to the optimal solution […] I’m not sure it counts as programming. Or, it was programming in action, but not coding, only input and output.

As a tool for analysis, thinking in terms of tactics does not provide a ready-made classification into which actions are to be grouped. Instead, it provides a frame of reference to express the ideas of autonomous agency within the strategic dimensions of power. Pursuing an interpretation of the teachers’ everyday practices in the realm of programming in mathematics might therefore demand a more nuanced language and more specific tactics.

5.7 Notes on trust and cumulative knowledge

Shapin (1998) would say that the modern scientist is a person of integrity, a person who would not misrepresent the truth. He refers to the way Boyle, who was a wealthy gentleman, would not have any hidden agenda to divulge the scientific method and this gave him some credibility. In this sense, the university as the ultimate research institution tries to reproduce the gentleman effect by founding research independently of commercial interests or at least publicly acknowledging what those interests might be. Openness, Shapin tells us, was from the beginning one of the foundations that were necessary for science to be an accepted way of reaching for knowledge. Openness—even if it was open to just an exclusive group of fellows in the Royal Society—made it possible to accumulate knowledge because it created trust. Once scientists believed they could trust previous results, they could build upon those without needing to replicate the experiments themselves (ibid., p. 108). Part of
this trust is also upheld by judicious statements, not claiming too much from one’s findings. The fact that fraud in research is so upsetting and defamatory is good proof of the value that we place on research integrity.

Here, one finds evidence, not only for the recognized moral value of not deceiving but also for an epistemological stance by which knowledge that counts is based on trust within the scientific community. That is, we know something, not because we have arrived at the conclusion ourselves, but because someone we trust has done so, and because the process that allowed that person to support her claims is made visible for the community and therefore susceptible to be inspected. In current publishing structures, we require not only the possibility of subsequent inspection but also a certainty that accurate scrutiny has been completed. This peer-review system moves trust one step further by increasing the size of the community members that stand behind the findings. Nevertheless, for the rest of us, trust remains the ground of knowledge for upcoming discoveries.

In social science, openness has yet another dimension, since the empirical data that might be retrieved is often context bounded and the results are difficult to replicate. A step in this direction would be to produce knowledge that avoids absolute claims, but instead progressively refines its narrative to elucidate the complexities of social experiences.

Showing that the results are valid across populations and settings outside those in the present study—the issue of generalizability—is a problematic matter in qualitative research (Maxwell, 2012, p. 141). Instead, the thesis strives for transferability (Polit and Beck, 2010, p. 1453 and Lincoln and Guba, 1985, p. 124) by focusing on logical argumentation and providing detailed descriptions of the empirical data. This information is expected to help the audience in their judgment as to whether or not the conclusions could be applicable to other similar contexts of interest. Nevertheless, the value remains in the genuine understanding of an educational setting within the described frame and a closeup analysis of one aspect of a nationwide curricular reform.

5.7.1 Ethical considerations

Ethical implications in data collection needed to be analyzed and considered against the overall goals of the thesis. Several premises were considered rele-
5.7. TRUST AND CUMULATIVE KNOWLEDGE

vant: information, voluntariness, confidentiality, and *primun non nocere*\(^6\). This section discusses the pertinent ethical concerns that could be relevant when gathering empirical data from teachers and in the presence of pupils in a classroom setting. The text includes a discussion of the potential repercussions of the results and an examination of plausible ethical dilemmas and considerations linked to the chosen data collection methods.

Maxwell (2012, pp. 100) writes that human relationships are particularly important in qualitative research and affect the data collection process, the empirical data, and the results. This includes not only the formalities around interviews but also respect and sensitiveness toward each participant and their personal situation. Thus, information was presented to the participating teachers both informally during the first contact meetings, and then formally with an introductory letter presenting the topic and asking for consent. The interviewees were notified about their right to interrupt any ongoing interview or classroom observation at any time and provided with contact information in case they wanted to withdraw their participation. At all times, the participating teachers were allowed to read the transcripts of their own interviews, with the purpose of commenting, rectifying, and even retreating from unintended utterances. The idea was to ensure teachers’ participative ownership of the material to honor their voluntary collaboration and as a form of data triangulation to remove possible misinterpretations.

Confidentiality requirements were resolved by coding the names of the participants in audio and transcript files. Their names might remain in the e-mail conversations that took place in order to arrange the meetings, but those names are not linked to their responses. Some meeting arrangements and even interviews were conveniently held over by different commercial platforms which are potentially available for third parties\(^7\). However, because there was no intention to gather sensitive information and the participants themselves were users of those platforms, it was resolved that the mere participation in the study, if somehow disclosed, would not represent a major inconvenience for a respondent.

The purpose of the specific data acquisition and the storage of the information was analyzed prior to the start of the project. Precautions were also taken

---

\(^6\)above all, do not harm

\(^7\)Skype, Facebook, Messenger, Twitter, etc
in order to ensure confidentiality, anonymity, restricted access, and timely re-
moval upon request (Dataskyddsförordningen, GDPR, 2016, Article 17, Right
to erasure). This is not only a legal matter but a question of trust. Informants
who have confidence in the data collection process would be more positive
when deciding whether or not to be part of a data collection for research.

Objectivity is both desirable and unfortunately unattainable for this kind
of social studies. The researcher’s deep involvement in the data collection pro-
cess makes observations and recordings percolate through individual layers
of theoretical inclinations, personal backgrounds, and chosen research plans
(Lofland et al., 2022; Hammersley and Atkinson, 2007). Honesty and trans-
parency regarding the background, would, if not prevent biased information,
at least expose it for external assessment.

5.7.2 Addressing positionality

The knowledge that qualitative research aims to provide is inevitably incom-
plete and situated. Information about the world is sieved through several
epistemic gaps before even reaching the researcher (Simandan, 2019) and a
few more on its way to publication. At this point of intersection, it is neces-
sary to revisit the author’s worldviews, ideas, and decisions about the way
qualitative research develops. This approach to positionality attempts to ad-
dress the fluid social and political context that creates each identity. It can be
constructed by bringing forward a person’s values and opportunities in terms
of macro-level sociological categories such as race, class, gender, sexuality, and
ability status. Ultimately, by describing how the author’s identity might have
influenced —and biased—, their outlook on the world (ibid.). Acknowledging
positionality allows therefore for a critical analysis of self-understandings and
the ways one recognizes reality through scholarly research and implications
on the choice of theories and methods. It’s time to nail my colors to the mast.

I have studied computer engineering and become a teacher in mathemat-
ics and computer programming. Approaching mathematics from a computa-
tional point of view is not alien to me, and I often resort to programming to
double-check algebraic solutions. I believe therefore that computational so-
lutions are a valid way to mathematical knowledge. This epistemic stance,
while not unique, positions my perspective on mathematics teaching on a
programming-positive side that is neither widely shared nor necessarily insti-
5.7. TRUST AND CUMULATIVE KNOWLEDGE

tutionally sanctioned. Nevertheless, this research does not aim to convince the reader about the benefits of programming in mathematics but to understand what makes policy blunter or sharper an instrument in changing mathematics teaching practice.

I am in this endeavor providing a transparent and self-reflective recognition of my shortcomings and biases that might complete the reader’s interpretation of the research presented in this thesis. This first-person positional- ity, however, should not be seen as an autoethnographic contribution, since it lacks answers to the motives and values behind my decisions.

Understanding the meaning and significance of a particular teaching practice, as it is understood by the teachers who engage in it, inherently favors an emic, insider approach. While it is challenging to portray the professional surroundings in which one was embedded, the accurate descriptions and concrete details are an attempt to help outsiders gain a richer understanding. The participants’ reflections, conveyed in their own words, not only strengthen the validity and credibility of the research but bring the teacher ‘dialect’ into academe.
Part III
Chapter 6

The strategic onset

The outcomes of the studies on teachers’ practice, when programming and mathematics converge, are presented in three consecutive chapters. This first chapter is devoted to the strategies. The tactical responses to the strategies are the matter of chapter 7 and finally chapter 8 elaborates on how tactics become accepted practices. The notions of strategies, tactics and practice refer to the regulative structures from Michael de Certeau’s opera prima, The Practice of Everyday Life (1984). Here, strategies and tactics are concepts used to illustrate the misalignment between people’s everyday ways of acting and the environment created by institutional goals. These two notions help to examine how teachers adapt their practices in response to the curricular revision while following their own pedagogical views of mathematics and programming.

The order of presentation echoes therefore an overall chronological precedence of strategies afore tactics afore established practice. Teachers’ tactical moves are interpreted as a —delayed— response to the strategic structures in which they operate. These strategic structures are multifaceted and could be examined from many different angles, each emphasizing different components and relations—cultural and historical developments, actors and activities, power relations, etc. For this analysis, however, the strategies exposed in the official documents will suffice to set the scenery that is needed to give depth to the stories in the foreground: those of the tactics employed by the teachers and the practice they construct with them.

The chapter starts by bringing to light the strategic decisions found in the
regulatory records that were consequential at some point during the period in which these studies were conducted (section 6.1). Bridging the official documents to the effect they might have had on teachers’ tactics, section 6.2 presents empirical evidence of a good understanding of what the first curricular amendment and later the syllabus revision entailed.

Within the respective sections, the developments are presented chronologically. However, both the precedence of strategies before tactics and the precedence of earlier events before later ones leave out the actual iterative process that resulted in the data that is to be presented. Some strategic sources were investigated after they appeared in tactical moves and some evidence of practice from the first iteration was first recognized after it was found again in the second iteration. This methodological itinerary was reported in more detail in chapter 5 and is not addressed here.

### 6.1 Strategies behind the new curriculum

The institutional texts that were produced along with the revised curriculum account therefore for the strategies that support the reform. In de Certeau’s terms, the strategic would be equivalent to the institutional, since strategy consistently emanates from a proper place, a place that only institutions hold (de Certeau, 1984, Chapter III). Therefore, it is in the written words that the inscriptive space is properly recognized and what constitutes the otherness confining everyday practices (see subsection 4.1.1). These documents originate from the explicit directives and ordinances that govern Upper Secondary Education (see Appendix 12.3), but also from the texts and short films that the Agency for Education provided as a support material to their teacher professional development initiatives and less explicitly, from the teachers’ experiences of the National Exams.

In this section, the perspective of interest when reviewing the official documentation is to elicit the intentions and expectations that are present in the formulation of the curriculum and shape its transformation and realization.

The documentation is hierarchically divided into three categories which reflect their legally binding effect and visibility (Figure 6.1). At the highest level, the curriculum for the upper secondary school and the mathematics subject syllabi (subsections 6.1.1 and 6.1.2) are described and analyzed. Fol-
6.1. STRATEGIES IN THE NEW CURRICULUM

Following that, the official comments to the subject syllabus are examined, along with subsidiary examples concerning the course materials and proposed classroom activities (subsection 6.1.3). There is a third more surreptitious body of text that has been proven to influence mathematics teachers’ practice: the National Exams. The strategies that this hidden curriculum actualizes with respect to programming in mathematics are presented in subsection 6.1.4.

This order represents a top-down classification regarding not only the legal affordances of the documents but also the abstraction of their implications. The course syllabus outlines general guidelines and a bare minimum to comply with, whereas the material from the official teacher training courses provides concrete examples and activities that pose a conceivable but not exclusive interpretation of the legal requirements. Finally, the material surrounding the National Exams can also be taken as cues that influence teachers’ practice.

6.1.1 Curricula for the compulsory school system

The overall purpose of the 2018 curricular revision in Sweden has been to promote a deeper understanding of how digital technologies affect society (Skolverket, 2017c; Skolverket, 2016; Skolverket, 2019a). Students should be able to use and understand digital tools and media, have a critical and responsible approach to digitalization, and be able to solve problems and convert ideas into action using new technology (Bourgeois et al., 2019). These ideas

Figure 6.1: Three levels in the inscriptional space conforming the strategies.
led to a broad revision of curricula to incorporate digital content creation and digital information assessment across all subjects from early school years as it was done at the time in many other European education systems (Bourgeois et al., 2019). To deliberately include programming in primary, lower, and upper secondary education, and to do so within already existing subjects was a far bolder move; a strategy that already indicated a strong belief in coding as a necessary competence to advance other knowledge areas and maybe the necessity to meet the digitalization requirements within the constraints of the actual teaching force.

The decision to integrate programming within already established mathematics courses had important implications for the teachers, who needed to coordinate the programming learning progression that their students would encounter. For this purpose, the Agency for Education explicitly mentions four abstraction steps adapted to the pupils’ level of cognitive development (Skolverket, 2019a; Skolverket, 2019b). The youngest children should therefore be gradually introduced to sequenced instructions and symbols as a way to express a structured process, for instance, to communicate a dance choreography or a recipe. In middle school (ISCED 1), pupils should be familiarized with more formalized algorithms to control digital devices with the support of visual programming environments. The last years of compulsory education (ISCED 2) should provide students with training in different programming environments and include it as a mathematical tool.

In mathematics, the subject syllabus for the compulsory school (ISCED 1, ISCED 2) stated in its first enforced version, that “through teaching, students will be given opportunities to develop knowledge in using digital tools and programming to be able to investigate problems and mathematical concepts, make calculations and to present and interpret data” (Skolverket, 2019a). In the current revision at the time of writing, the requirement has been gathered around the knowledge topics for Algebra. Problem-solving is no longer listed as a mandatory area where programming must be used. The degree of difficulty has been lowered and is now formulated as “Simple programming in visual programming environments” but the specific passage about “how algorithms can be created, tested and improved in programming” remains (Skolverket, 2022e). These changes occurred in parallel with adjustments in the subject of technology, where programming is now more clearly linked
6.1. STRATEGIES IN THE NEW CURRICULUM

to automatic control methods for robotics applications rather than computational software.

Despite retrenchment, the curriculum still compels teachers to train the students in programming in some way. Which tools to use or which programming language to teach are choices left to be made locally and it is the teachers’ responsibility to select appropriate activities for their students to learn computer programming. This strategic decision has significant consequences for secondary and higher education, teacher training, and student mobility within and across educational systems.

6.1.2 Curricula for the non-compulsory education

Below the Swedish Education Act, the Upper Secondary School Ordinance is the overarching legal frame that applies for Swedish ISCED 3 levels. The revisions discussed in this thesis are therefore the different amendments to this ordinance. The dispositions regarding the mathematics curriculum are of particular interest for programming in Upper Secondary Education.

The ordinance for non-compulsory education prescribes that mathematics is a core subject in all ISCED 3-programmes (Skolverket, 2019b). The different subject syllabi of mathematics at this level follow a common kernel in all the courses while the actual contents are adapted to the particular needs of each vocational field.

In the curricular revision of 2018, programming became a part of the mathematics course syllabi at all levels in the technical and science programmes and appears first in the third course of the economics and social science programmes (course 3b, see Appendix 12.3 for an overview of the Swedish school system). The programming requirement was enunciated with the same sentence independently of the course. The mathematics curriculum established that “The teaching activities in the course must include the following central contents: […] methods for mathematical problem-solving, including modeling of different situations, both with and without digital tools and programming” (ibid.)

Soon after, in 2021, the curriculum was updated again. In this second revision, the programming requirements were downplayed and stratified, with

12018: Strategier för matematisk problemlösning inklusive modellering av olika situationer, såväl med som utan digitala verktyg och programmering.
lower demands in the first years. In the introductory mathematics courses, teachers should provide only “examples of how programming can be used as a tool for problem-solving, data processing, or application of numerical methods”. The advanced courses’ syllabi indicated that not only examples but also use of code in those same areas should be available among the teaching activities (Skolverket, 2022c). In the new course syllabi, programming is still presented as a way of working with other mathematical content and it is included in the same category as other digital methods and tools to streamline calculations but modeling and simulations are no longer expressly mentioned (Skolverket, 2022d, p. 9).

The second version of the curriculum for upper secondary school keeps the role of programming among other problem-solving techniques and omits any explicit progression regarding programming. This does not exclude an implicit progression in the core mathematical content of each course which could frame the kind of data processing or numerical methods that are to be exemplified with computer programming. In conclusion, programming is necessary as a general method, but no particular programming paradigms or languages are mentioned. The curriculum does not specify which data structures the students will need to master nor does it mention how to bring the students up to speed with the necessary programming skills to simulate, model, and solve problems.

The mathematics curriculum has undergone two revisions in a relatively short period, first in 2018 and then in 2021. Furthermore, in 2022 the Agency for Education announced its intention to implement yet a new revision in 2025 (Regeringskansliet, 2023). This forthcoming revision will affect all the subjects in the Swedish ISCED 3 and introduce significantly more extensive modifications compared to the previous revisions. In mathematics, the courses will be merged into fewer but larger subject segments (Skolverket, 2022a). With respect to programming in mathematics, the proposal reproduces the same excerpt “examples of how programming can be used as a tool . . . ” that was already in the 2021 version of the curriculum (ibid., p. 5 and 8). Table 6.1 presents a chronological summary of the regulatory decisions that were of consequence for the implementation of the reform in ISCED 3.

---

2022: Exempel på /Användning av hur programmering kan användas som verktyg vid problemlösning, databearbetning eller tillämpning av numeriska metoder.
6.1. STRATEGIES IN THE NEW CURRICULUM

Table 6.1: Timeline for the regulatory decisions concerning programming in ISCED 3.

<table>
<thead>
<tr>
<th>date</th>
<th>Resolution</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sept 2015</td>
<td>The Agency for Education shall propose the necessary changes in curricula and syllabi to reinforce and clarify programming as an element of teaching.</td>
<td>Cabinet order U2015/04666/S Commission to propose national IT strategies for the school system. (Utbildningsdepartementet, 2015)</td>
</tr>
<tr>
<td>Oct 2017</td>
<td>Students shall understand how digitization affects the world, how programming controls both the flow of information we access and the tools we use, as well as how technology works.</td>
<td>Cabinet order U2017/04119/S National digitization strategy for the school system. (Utbildningsdepartementet, 2017)</td>
</tr>
<tr>
<td>July 2018</td>
<td>The mathematics curriculum including programming enters into force.</td>
<td>Ordinance 2022:12 Amend on SKOLFS 2010:261 on subject plans. (Skolverket, 2017b)</td>
</tr>
<tr>
<td>July 2022</td>
<td>The second revision of mathematics curriculum for ISCED 3 enters into force. Programming is downplayed.</td>
<td>Ordinance 2022:12 Amend on SKOLFS 2010:261 on subject plans. (Skolverket, 2022c)</td>
</tr>
<tr>
<td>July 2025</td>
<td>The subject reform will enter into force. Grades are to be given for subjects rather than for courses.</td>
<td>Ordinance 2022:12 Amend on SKOLFS 2010:261 on subject plans. (Skolverket, 2022a)</td>
</tr>
</tbody>
</table>

By framing programming within mathematics, rather than an independent science, the documents revealed a strategy that goes beyond the explicit directive and encroaches upon the identity of both subjects. By framing programming knowledge as a tool (swedish: verktyg) the syllabi levels coding with other tools used in education, such as measuring instruments or a dictionary. This is a crucial strategic choice because of its capacity to shape the teaching activities and the TPD that take place but more importantly because of the ontological consequences for mathematics teaching practice. There are reasons to believe that tools in mathematics do not enjoy the status of core knowledge or even necessary skills.

Pupils might have once learned how to draw circles with a compass to find the perpendicular bisector dividing a line segment into two equal parts (Figure 6.2). Now the syllabus requires algebra to find the exact coordinates of the midpoint given two endpoints, but the compass itself did not make it into the modern classroom. It is geometry that belongs

Figure 6.2: Bisector of a line segment.
in mathematics, not the drawing tool.

This distinction lies at the bottom of the argument when the dichotomy between technology and practice is discussed later in chapter 8.

6.1.3 Programming activities and instruction examples

This second category of data consists of a much larger corpus of semi-official documents whose goal was to clarify and inspire mathematics teachers in their endeavor to learn and apply programming in their mathematics teaching. Here we find a publication with comments on the new curriculum (Skolverket, 2019c; Skolverket, 2022d) as well as a set of activities and programming examples ready to use in specific courses (see Table B.1 for a summary of activities endorsed by the National Agency for Education). The available records regarding invitations to tender for teacher training courses commissioned to universities are also considered, particularly in relation to contents and progression prerequisites (eg. Statens Skolverk, 2017).

The material for upper secondary school is centered around text programming with a clear inclination to imperative paradigm (mostly in Python) although functional programming examples with Wolfram Language (Wolfram, 2016) are also provided, and even solutions with spreadsheet applications. While there is no intrinsic order among the proposed programming activities, they target specific mathematics topics that can be linked to different courses and their inherent progression. When compared to the level of difficulty in the code as described in section 3.2, it becomes apparent that there is no expectation of providing progressively more demanding exercises in higher courses. First-year students are presented with programming activities requiring the use of external libraries, nested constructs, advanced formatting, and even access to the computer’s file system, well above the level of difficulty of some of the programming activities proposed for later courses. Because of the variety in programming paradigms, data structures, and computational complexity found in the examples endorsed by the Agency for Education, it would be easy to believe that there is no upper limit for how many advanced programming features can be embedded in the exercises as long as they convey the mathematics topic to be studied.

Although the National Agency for Education is behind the texts and video material analyzed in this section, there are still some disparities in the over-
all strategy. This becomes apparent in one of the lesson examples produced as part of a competence development effort in mathematics didactics (NCM and Skolverket, 2019). During the interview following the lesson, the teacher makes a point of not using programming for purposes where other computer applications would render more powerful. They give a specific example of calculating integrals numerically, which they mean should be handled with other digital tools better tailored for that purpose. Programming, they mean, is to be used as a problem-solving tool, and not necessarily to improve conceptual understanding.

This view is challenged in several of the activities that the same institution offers, where the explicit purpose is to explore mathematical concepts and expand students’ understanding of procedures rather than to solve a particular problem.

For example, parting from the code in Figure 6.3 used to solve systems of two equations, the students are asked to explain how the program works and how it could be modified to solve systems of three equations or quadratic equations (see Table B.1).

```
# ekvationssystem med två okända

a1 = float(input("a1?"))
b1 = float(input("b1?"))
c1 = float(input("c1?"))
a2 = float(input("a2?"))
b2 = float(input("b2?"))
c2 = float(input("c2?"))

if -a1/b1 * c1/a1 == -a2/b2 and c1/a1 == c2/a2:
    print("Endligt många lösningar")
elif -a1/b1 == -a2/b2:
    print("Lösning: x=
else:
    x = (b2 * c1/b1 - c2)/(-a1 * b2/b1 + a2)
    y = -a1 * x/b1 - c1/b1
    print("Lösning: (" + str(x) + ", " + str(y) + ")")
```

**Figure 6.3:** Code example to solve systems of equations with two unknowns (Helenius and Dahlberg, 2018).
CHAPTER 6. THE STRATEGIC ONSET

This activity is intended to increase students’ ability with general algebraic solutions and it is well suited for programming beginners. However, the cognitive difficulty that the students face lies in finding the algebraic formula for the solution and this needs to occur before any coding can take place. This means that the real problem needs to be solved before it can be programmed. Therefore, the activity could rarely be considered an example of “using programming for problem-solving”. It also contradicts the idea of using the most appropriate digital tool because coding the solution from scratch would seldom be the method choice when trying to solve equations.

Other official records were gathered from the National Agency for Education Syllabus model for the programming courses commissioned to universities (Figure 6.4). Among them, there is an explicit and detailed description of the programming content that in-service teachers should learn in these teacher training courses. The central concepts in the imperative programming paradigm are listed in the usual order—sequence, alternative, conditions, loops—, alluding to a reasonable understanding of increasing difficulty. If in-service teachers are taught programming under this assumption, it is sensible to believe that they might replicate that same arrangement with their own pupils.

Figure 6.4: Syllabus model for the programming courses for in-service teachers commissioned to universities (Statens Skolverk, 2017)

To summarize, the manifestation of the curriculum in comments and examples is in itself a strategy from a position of power; it narrows the interpretation of the official curriculum, without enforcing it with more strict measures that would imply diminished acceptance among practitioners and would re-
6.1. STRATEGIES IN THE NEW CURRICULUM

quire a purposeful commitment to teacher professional development. This is a double-edged sword: catering to both those who value flexible guidelines and those who welcome the guidance of normative examples and at the same time deploying reasonable resources into the enterprise. On the downside, it leaves the teachers without any assurance regarding what kind of programming skills to learn themselves and what to expect from new students. This opens for larger knowledge spread among students and precludes principals from soliciting funds for training purposes since there is no solid regulation stating a bare minimum of programming qualifications to be met.

6.1.4 Strategies in the hidden curriculum: the National Exams

The official narrative is explicit in the curriculum and related documents but those are not the only texts guiding and constraining mathematics teachers’ practice. A powerful strategic device that is often mentioned in the discussions is the National Exams. These national course tests in mathematics are developed based on the objective and criterion-referenced syllabi. The tests are broadly used in Swedish upper secondary schools and the obtained results shall be reflected in the individual grades. Not surprisingly, officially sanctioned large-scale course assessment defines what counts as a valid demonstration of knowledge.

During the interviews, there were abundant references to the National Exams and their influence on the decisions that the teachers made regarding the topics and the methods to be deployed when teaching mathematics. The content of the National Exams is classified information for seven years after the date the exam was last used. This means that there are currently no publicly available exams from the period since programming was introduced in the curricula. Therefore, the claims that the teachers make regarding programming being absent from official assessment cannot be demonstrated with actual examples but there are pieces of information that corroborate their views.

Firstly, we have access to the formula sheets that the students are allowed to use in the National Exams. These memory aids provide an overview of formulas, key equations and definitions relating to geometric measurement, alge-

---

3Criterion referenced assessment is the process of evaluating (and grading) the learning of students against a set of pre-specified qualities or criteria, without reference to the achievement of others (Brown, 1988; Harvey, 2004).
CHAPTER 6. THE STRATEGIC ONSET

bra, and calculus concepts. The official formula sheets are regularly updated and released by the Agency for Education, keeping up with the changes in the respective course syllabus. They represent a further concretization of the statements in the syllabi, by translating sentences such as “Handling trigonometric expressions, and proof and use of trigonometric formulae” (Skolverket, 2019b, p. 35) into the specific set of trigonometric relations that can be expected to be useful in the exams (see Figure 6.5). Therefore, the choices that the Agency for Education makes when selecting the entries on their formula sheets are closely monitored by teachers and students in search of cues about what to expect in the tests.

Figure 6.5: Entry on the official formula sheet enacting the preferred interpretation of the requirement “Handling trigonometric expressions, and proof and use of trigonometric formulae.”

The students writing the National Exams in mathematics are allowed to use digital resources for the second part of the test. These technologies can vary widely among schools and guidance regarding their use is therefore outside the scope of the formula sheets. Having entries with programming significance among the formulae would definitely indicate that programming was tested. This could potentially unveil what kind of programming knowledge the students were expected to be acquainted with, at which level, and maybe even a favored programming language. Yet, the absence of such information in the formula sheet implies that the use of programming is at most discretionary. Furthermore, it aligns with a view of programming as a tool rather than a core mathematics practice (see chapter 8).

A second hint is found in a proposition in which the government called for the national exams to become fully digital by the year 2022 (Regerings-
Unfortunately, the outcome of the Schrems II case notably delayed the time schedule for the digitalization of the National Exams, which are now planned to be effective by the end of 2025. As long as sound technical solutions are not implemented in all schools, it will be unfair to include test questions in which programming would make a significant difference. The implementation of a digital solution to National Exams will coincide with the announced third reform of the upper secondary school curriculum. For mathematics, the reform would bring larger evaluation periods that will allow the students to immerse themselves in the subject before the final grade is set. This will necessarily have implications for assessment. By then, the digital solutions shall be ripe and the student cohort would have encountered programming in school for at least eight years. This might be the occasion to open for programming exercises in the National Exams. Until then, it is reasonable not to expect any drastic changes in the way those tests are constructed regarding programming tools, devices, and contents.

The two developers of the national tests under the aegis of the Swedish Agency for Education confirm that so far, there has been no task in the national tests that required students to understand code or to be able to program. According to PD2:1 this is partly because there is no canon about which programming language(s) can be used, and partly because programming on paper is not something that the Swedish National Agency for Education wants to emphasize (PD2:1).

With the contents that the courses cover, there are not that many natural parts where programming would be the first choice. If you have access to a graphing calculator, GeoGebra, Desmos [interactive mathematics environments with dynamic geometry] or other adequate tools, the need for programming is not that great. It is faster and easier to just use the tool. In the student solutions that we have collected from the national tests, we have not seen any student who presented a programming solution [...] It is a bit tricky, I think, that programming is added to mathematics at the same time that the numerical methods are removed or pushed to the last courses.

4On 16 July 2020, the CJEU issued the Schrems II judgment with significant implications for the use of cloud services that were essential for the implementation of digital national exams.
CHAPTER 6. THE STRATEGIC ONSET

The university group in charge of the National Exams for younger students (ISCED 2 and the first year in ISCED 3) acknowledges that so far, their trials to include programming exercises in the test have not been successful and they mention immature technology in some schools as well as teachers sometimes scarce knowledge of programming. Another difficulty comes from the lack of agreement on which programming language the students can be presumed to know.

Before we had the topic of integer divisibility, which presented programming opportunities together with problem-solving. Now those are gone, and our target group [first year ISCED 3] is only supposed to see examples of programming. This makes it even more difficult to find a good entry point

There is no official requirement for all the goals on a course syllabus to have a bearing in the National Exams and there are other precedent examples of syllabus content that has escaped the tests. For instance, the rubric “Teaching in the course should cover […] mathematical problems related to the cultural history of mathematics”(Skolverket, 2019b) is not being assessed in the tests either, despite it being part of every course in the subject. Between 2011 and 2021, the cultural history of mathematics was part of the core content but also an explicit knowledge requirement, which has never been the case for programming. Still, and probably because here too there is no canon to which problems to include, the assessment of whether students could give relevant examples that related other course content to the cultural history of mathematics was left to the teacher.

Two important strategies can be gathered from the absence of programming in the national exams. First, it reiterates the idea of programming as a tool among other tools, overlooking its affordances above simpler digital devices such as calculators. The second strategy aligns with the resolution of introducing programming simultaneously in all levels, from ISCED 1 to ISCED 3, which presumes a long gestation period until both teachers and students are up to speed. During the first years test developers are left in the no man’s land where programming is neither fish nor fowl, not to be forbidden,

---

5This requirement has been removed in the current version (Skolverket, 2022c) coinciding with the requisite that the student’s results on the national exam must be reflected in their final grade.
6.2. FROM STRATEGIES TO TACTICS

not to be required, not officially discarded but yet unrealized. This is a perilous decision that might compromise the intentions of the government’s bill for fair, equal, and digital National tests (prop. 2017/18:14; U2017/03739/GV, Regeringskansliet, 2017b). It is unreasonable to require programming skills as long as not all the students have access to programmable devices during the exam, but is it fair that those who have access to this powerful tool are allowed to do so and still be judged by the same standards?

6.2 From strategies to tactics

The policy documents in effect are the official strategies that move the positions forward toward the governmental ambitions. The strategies endorsing the new curriculum are accessed here through the official documents analyzed in subsections 6.1.1, 6.1.2 and 6.1.3, such as the revised national curriculum itself and the support material produced by the Swedish Agency for Education (Skolverket, 2019b; Skolverket, 2022c; Skolverket, 2019c), as well as the programming workshops and the courses commissioned to universities and the National Exams.

This was therefore the first step to operationalize the dichotomy between strategies and tactics outlined in subsection 4.1.3. The response to these strategies is to be found in teachers’ individual efforts to incorporate computer programming into their mathematics courses while tactically allowing their own beliefs to shape the teaching activities in their classrooms. This will be the scope of chapter 7. To bridge the gap between the aforementioned strategies and the expected tactics, it is necessary to understand to which degree the institutional texts reached the practitioners interviewed in this study and whether the strategies that they represent were purposely recognized. As de Certeau notices, “power is bound by its very visibility” (de Certeau, 1984, p. 37). The efficiency of the strategies in place relies on the same hegemonic structures that perpetuate power, and so is the alignment of tactics and later practices. Discernible evidence of the impact and transparency of the curriculum revision and related production is therefore reported in this section in two chronological subsections that correspond to each of the iterations in the investigation.
6.2.1 The significance of policy during the first iteration

Before programming was officially introduced in the mathematics curriculum in 2018, the Swedish Agency for Education had already prepared the ground with programming workshops and other professional development activities for teachers and school principals. The reform seems to have been well announced ahead of its implementation, and none of the teachers mentioned to be unaware of the changes.

I wouldn’t say we were well prepared, but we knew that programming was coming. I had a teaching intern who was a former engineer, and we discussed programming quite a lot, trying to find good entry points, […] and [name], my colleague, was in one of the workshops but I think they came back more scared than inspired!

With regard to the curriculum amendment and its interpretation, there are several references in the excerpts from the interviews that shed light on this matter. For example, in the following passage (A:2), Ariel is quoting the text in the official document and is therefore acknowledging its existence and relevance for the choices they make in their teaching practice.

This is, the way I see it, an example of “mathematical problem-solving with programming” but it’s not necessarily the way I would choose to solve this problem

Ariel expresses skepticism towards the reform by immediately objecting that but it’s not necessarily the way I will choose to solve this problem. By conducting a teaching activity in which they do not fully believe, Ariel shows a tactical behavior that checks off one box in the curricular requirements and at the same time allows them to pursue what they mean are more relevant issues.

In another reference to the curriculum, Ariel mentions a separate programming course as a preferred alternative to increase students’ coding skills (A:3). While “the ambitions that come from the new curriculum” is a vague statement with respect to what those do entail, it is nevertheless a recognition of a change toward a more demanding task. The second idea in the passage about ‘making programming class compulsory’ discloses a comprehensive knowledge of the upper secondary school curriculum and the existence of a yet elective programming class.
Mathematics already contains a lot. I was a little worried about, and I am still a little worried about how to fulfill the ambitions that come from ... in the new curriculum. [...] I'd rather see that the subject of programming was made compulsory, at least for these students, in the C-line [technical and natural sciences programmes, (see Appendix 12.3)]

There are various references to other practicalities of the reform in the data. For example, Dominique’s critical note on the curriculum revision alludes to the simultaneous implementation in all courses (D:1). While there is some dose of resignation in the episode, there is also an overt tactic of contention or harnessing, that serves the goal of using resources wisely. Both Dominique’s planning time and the students’ time are scarce, hence, first-year students are prioritized, because with them, the rewards of such an effort could be harvested later.

_I have these students, [in reference to the first-year group that had just left the classroom] and I’m going to be their teacher for three years, and that’s an advantage, because programming needs to take some time so they can improve, so they can see that it’s useful, but for the other course, they have one month left, and we haven’t really done any programming yet. I’m not gonna start with programming now, we don’t have time before the National Exams, and they will most likely disappear after that [...] I’ve thought about it, because the course includes solving trigonometric equations with numerical tools, so maybe, I thought, that would be a good place to program, but at the end, I just followed the unit plan as I use to. It wasn’t worth it to start changing. But next time, maybe, in two years, then the threshold won’t be so high, neither for me nor for these students._

In the following passage (H:1), Hanne provides two important cues that can be recognized as broad knowledge of the policy documents, including those outside their immediate professional scope. There is not only a reference regarding what is expected from mathematics teachers in upper sec-
secondary school but also one about what is expected from teachers in compulsory school. This understanding grants awareness of the progression that the official documents mandate.

We are not supposed to teach programming here, they should know it already and maybe, in five to ten years, then we have students that have had programming all along and could write code themselves, but for now, I just give them glimpses of what is possible, because the course is full as it is.

Other teachers made similar observations in reference to the new directives. It was common to point out that teaching programming was outside the scope of what mathematics teachers in upper secondary school were required to do. In the following passages, both Gaby and Hanne make related remarks about what to expect from students entering upper secondary education (G:1 and H:1).

The idea was that the students will get it with them more and more [programming knowledge] from compulsory school, but just that in the first years they do not have it, they have no knowledge at all most of them.

The second source of strategic documents were the programming activities and instruction examples suggested by the agency for education (in 6.1.3). In one case, the transcripts provide direct evidence of those documents having some influence on teachers’ practice. Evin recalls some of the material produced for the programming training courses that were offered during the previous year and extrapolates the algorithm to the topic in their current unit plan (see Table B.1).

Just test a bunch of exponents until you are close enough, you know, like in the one about approximating the number $e$ [in reference to an exercise found in the workshop material]

Evin seems to draw inspiration from the exercises that were available at the moment and compares their approach to calculating logarithms with incremental trials to one of the exercises presented in the first workshops (E:1). The resemblance with an exercise that was approved by the Agency for Education serves to justify why the programming activity was selected for one of
the lessons. The tactic, in this case, could be an example of *la perruque* (4.1.4), in that a presumably valid form of ‘programming within mathematics’ is subverted to convey a concealed purpose. The activity is adapted to a different topic to suit the needs of the practitioner—to teach about logarithms—rather than to apply programming to solve a problem. The difference is nevertheless subtle; while the original exercise instigates students to find a way to approximate a numerical constant, Evin wants their students to understand a concept: what it means for a number to be the logarithm of some other number in a certain base. The students are provided with the solution to the approximation problem (see Figure 6.6) and they can use it to explore under which conditions it is possible to find a logarithm. Here one can see how this tactical making-do is merely a semblance that appears to conform with the strategies. It does so, at least on a formal level, but diverges from the intentions of the curriculum to use programming for problem-solving.

![Image](image.png)

**Figure 6.6:** Evin’s code for incremental approximation of a logarithm (2020) \( \log_{10} 2 \)

Some of the official materials that were analyzed in subsection 6.1.3 might have remained unnoticed, probably because those exercises were intended for
training purposes, that is, for teachers learning to program, which were not represented in this study. However, the teachers did not just paraphrase policy texts but also remarked on the absence of regulation pertaining to how programming is to be implemented in mathematics. They were aware of the wider context of the reform, including how it is organized in compulsory school and in other subjects. Therefore, it is safe to say that the details of the reform were well-known among the informants even if the strategies that they represented remained undisclosed.

6.2.2 The significance of a new policy in the second iteration

The second iteration of interviews took place shortly after the Swedish Agency for Education revised the mathematics curriculum in July 2021, compelling the teachers to readjust the content of their courses. In the new curriculum, some topics had been reallocated to another level, some topics had been dismissed and new content was now part of the syllabus. Along with these adjustments, the programming requirements had also been modified. Programming took a less prominent role in the first courses and had consistently remained excluded from the National Exams (subsections 6.1.2 and 6.1.4). Teachers’ awareness of these changes could be observed in the empirical data, both in the unit plans that the teachers provided and in their comments during the interviews. While the diminished programming requirements were generally noticed, they did not necessarily impact the teaching activities or the tactics but rather the incitement to pursue them.

Before it was ok to be pushing it, we were few and the students got it, it was the novelty of it. Now I’m a bit wearier with the whole thing. After 5 years programming has not appeared in the National Exams, not once, which in retrospect is understandable, it couldn’t have gone any other way. But back then we still believed we were giving our students some advantage. and I still believe it, but I’m not excited about it

In reference to the new course syllabi taking effect in 2021, Hanne gives a short account of the consequences of readjusting the content of the courses (H:2).
6.2. FROM STRATEGIES TO TACTICS

I get why, the gap between courses 1 and 2 was a problem for many, so some refurbishing was necessary, but now [...] because before we had content-packed courses but at least some of the stuff was repetition, so students could get a much-needed respite. Now [the course] is content-packed and all new stuff [...] a bit less focus on programming but I don’t think about it that much, whether the thing [the programming activity] is “using code” or just “an example of code”, it’s a subtle difference.

Hanne continues by exemplifying how the changes have affected the programming content in their courses with the example of base conversion. They too mention the National Exams as being constraining on the way that the course is tested, and by extension, on the way it is taught.

The examples about converting from a number in base other than ten or to base other than ten, those were nice to program. The typical boring repetitive calculations were just cut out for it. Now they are in course 5 rather than 1. I think is better, it fits well there with the rest of the discrete mathematics, and there is no National Exam for that course, so more freedom in how to test.

The difference between course grade and subject grade, which is at the core of the 2025 reform was only sparsely commented on by Ariel. It was followed by a more significant remark about the essence of mathematics teaching practice beyond the particularities of a specific curriculum.

The courses are changing all the time nowadays, they want to sell more books, haha, but seriously, you have to find your ground, you have to know what you are doing and why, and hopefully how too. So I think programming is a good approach in mathematics, if you understand how it belongs, then it doesn’t really matter how much of it is explicitly stated in the course syllabus or subject syllabus, you just use it when it fits. So, yes, I will keep programming in mathematics, as long as they don’t take away the computers altogether. Designing an algorithm that solves a problem will always have more value than executing the steps of a process that someone else has come up with.
CHAPTER 6. THE STRATEGIC ONSET

There is therefore evidence that the second curricular changes had repercussions in the teaching activities that included programming. This needs to be understood in the context of the second revision of the curriculum but also in light of the consequences of the National Exams — the hidden curriculum that was explored in subsection 6.1.4. There is a clear trend of programming taking up less time than it did at the beginning but also deeper insights on how to balance the external requirements with the personal understanding of the teaching practice.

6.3 Summary of the strategic onset

The chapter was dedicated to the strategic onset, describing the forces that governed programming teaching in mathematics in Sweden at ISCED 3 level. Those included two curricular revisions, official initiatives towards teacher training and related material to support classroom activities, and finally, the tacit decisions exposed in the National Exams.

After revising the official documents, an overarching strategy can be outlined; one which prompts teachers to include programming in their mathematics teaching by means of examples related to the core matter of the course to the extent they consider manageable.

The significance of the strategic onset among the participant teachers was later discussed and supported with evidence from the interviews. The purpose of these two sections was to bridge the epistemological gap that separates strategies and tactics into two different but interdependent realms of action. It is by acknowledging the existence of strategies that adaptive practices can be understood as tactics. Correspondingly, it is because the teachers are aware of the changes in the curriculum and its ramifications, that they adapt their teaching and that these adjustments can be explained in terms of tactics. This will be the mission of the following chapter.

To explore how these strategies played out in the transformation arena (see Lindensjö and Lundgren (2000) in section 2.2), the following chapter is devoted to teachers’ practices when programming and mathematics converge. The empirical results are reported and analyzed to find the emerging tactics shaped by the institutional strategies which can be illustrated with examples in two different categories: Dual Teaching and Interspersed Programming.
Chapter 7

Tactical responses

After clarifying the strategies in place, this chapter describes the first contribution of the investigation: the disclosure of two different tactical approaches to the integration of computer programming in mathematics. These tactics are rooted in two different ontological ideas of learning programming and mathematics: *Dual teaching* and *Interspersed programming* which are further described and exemplified in sections 7.2 and 7.3. Together, the empirical results elicit a comprehensive and refined comparative description of the tactics summarized in section 7.4 that will delve into a broader analysis of practices in chapter 8.

7.1 Teaching when programming and mathematics converge

At the onset of this investigation, the introduction of programming in upper secondary school mathematics was a recent event and research about how to integrate the subjects was scarce. In an attempt to understand how teachers used their knowledge of both programming and mathematics in their teaching, the Knowledge Quartet framework (section 4.2) was initially considered. Particularly one of its four categories—connection—was expected to provide some insight regarding how teachers’ combined mathematics and programming knowledge was evidenced in the learning activities that they designed. At the bottom of this approach lay the idea that the subcategories *decisions*
about sequencing and anticipation of complexity would prove useful in outlining a learning progression for the joined field of programming in mathematics.

Once the episodes extracted from the interviews were tentatively pooled into the scenarios that define the Knowledge Quartet, it became clear that there was an underlying rationale that dodged the framework with respect to the connection category. For example, the contributory code of decisions about sequencing was being populated with comments about teaching programming such as Ariel’s mention of next step A:1 or when Chris declares that loops were new (C:2).

In passage B:2, Billie talks about how the class would start with arithmetical operations. This information could at first glance be interpreted as mathematics content since arithmetic is a notorious branch of mathematics. However, arithmetic operations are well known for students in upper secondary education and their study is not a part of the syllabus of the courses that Billie was teaching at the moment. For this reason, it is reasonable to believe that it was only the programming aspect of arithmetical operations that was being sequenced with regard to cognitive complexity.

Dominique’s observation that programming helps understanding, for certain situations that are not so obvious right away would qualify as an example of anticipation of complexity. However, it is the abstract probability concept, in the mathematical domain, that was expected to be difficult, disregarding any possible struggles with programming.

Evin’s comment about programming logarithms (E:2) conveys what could be expected of an integrated programming and mathematics decision about sequencing. The point of departure is the mathematics topic—logarithms—which is sequenced within activities of increasing complexity in the unit plan. At the same time, Evin consciously caters to a varied spectrum of students with different skills which surfaces when motivating that it can be adapted to how much programming the students know. Because that can be very different.

Ultimately, it was too rare if not unique to find episodes featuring the same category in both knowledge domains, such as Evin’s example in E:2. In general, for the programming exercises and examples provided by the teachers, the Knowledge Quartet did not result in a productive description of a converging mathematics and programming content learning. Fortunately, it did open a new path of analysis grounded in teachers’ different points of depar-
ture when planning for an integrated mathematics and programming curriculum. After all, there is an undeniable distinction between *planning mathematics activities with elements of programming* and *planning programming activities with elements of mathematics*.

### 7.2 Dual teaching

When adapting their lessons to the new curriculum, some teachers opt for separate instruction, in which some lessons are dedicated to computer programming, whereas the rest of the course remains more or less as it was before the reform. The term *dual teaching (DT)* is purposefully used here to emphasize this partition. The lessons dedicated to computer programming start with a basic presentation of a programming environment and build up according to a pedagogical view of increasing difficulty. New programming concepts are introduced gradually, and the students are given exercises to code at each level. The students are often already familiar with the mathematical topics that are demonstrated when programming is on schedule which allows them to focus their attention on the coding aspects that may be new for them.

In the following excerpt from Billie (B:1), we can see that programming is explicit in the unit plan, but also that it is considered less important for the students.

Billie shows a unit plan for the four weeks after the winter holidays. Some handwritten arrows indicate a lesson that needed to be moved to a later slot. “*My kid was sick that whole week, so I needed to rearrange some of the content*”, the teacher excuses. “*Normally students need to follow the plan by themselves if I’m absent. You don’t get a substitute for just a couple of days here […] that’s why they get these detailed plans with the exercise number and that, so they can work on their own.*” Nothing more is said about the adjustment and the interview moves on to talk about the previous programming lesson.
Here we find an example of a tactic hidden in the speech that is made apparent by analyzing the accompanying material, as it was justified in 5.3.3. It is a tacit tactic concealed in the spoken interview but revealed in non-spoken information.

Later analysis of the unit plan showed that the teacher was absent for two lessons in which the plan was to introduce the unit circle as a part of the trigonometry topic. The teacher had then squeezed these two lessons into a single one on Monday the week after, probably because the content was too difficult for the students to handle alone. What was planned for that Monday took place on Wednesday, and Wednesday’s activities were moved to Thursday. Then the chained lesson shift ended and a lesson initially dedicated to programming is dismissed in that unit.

Billie could have made a decision about which content the students would be able to learn on their own and which content needed to be explained in class. In reality, the mathematics content is honored on time whereas the programming lesson is disregarded until further notice.

This behavior is representative of a dual teaching approach in that it makes a clear distinction and priority among the two subject matters. Billie has most certainly the students’ best interest at heart particularly considering the tight schedule and the imminent trigonometry exam that the students need to pass. The adjustments made to the unit plan due to the contingencies of the short absence are both a sacrifice and a subtle tactic. Billie makes-do with the resources at hand, a tactical type of operation that de Certeau recognizes as bricolage (subsection 4.1.4). It represents a creative arrangement made with the available materials in which the teacher “readjusts the residues of previous construction” (de Certeau, 1984, p. 174).

In the following episode, we see how programming is used tactically to achieve several self-defined goals in the contingencies of teaching practice. In analyzing the example using the Knowledge Quartet framework, Chris demonstrates effective teaching practices (C:1). de Certeau’s theory adds insights about the tactical use of programming, the layer of the real in Critical Realism (see p. 51).
Half of the girls in the class were out with the school’s choir, it was Lucia, and the rest of the students wanted to do something Christmassy. […] Someone proposed to organize a Secret Santa, and sure, we could do that, but I said that it needed to include even those that were out that day, and how do we do it, so that nobody needed to give a present to themselves? and they ’It won’t happen!’, so I pushed like, ’won’t it? what’s the probability for a random list shuffle to produce a list in which at least one of the elements has the same position as in the original list?’, I started to show how one could calculate that probability for a short list, say 3 names, but they weren’t up for any serious math by then, and even less so for off-syllabus math. But then I programmed a simulation to see how often that happened. […] Just me programming, they hadn’t brought their computers, but that way I introduced lists, a least superficially, which was smooth, so we could take it from there next time we had programming.

If analyzed with the categories in the Knowledge Quartet framework, one could find in this example (C:1), how Chris thoughtfully instantiates ‘deviation from agenda’, ‘responding to students’ ideas’, ‘use of opportunities’ and ‘teacher insight during instruction’, all belonging to good professional practices in the contingency dimension. It is in that respect what could be expected from exemplary teaching. With de Certeau’s theory, new insights complete the narrative. Here, it is possible to see that the teacher resorts to programming as a leisure exercise, a makeshift lesson—again a tactic of bricolage—, reifying what is given to fulfill one’s purposes. Chris takes advantage of the situation to advance the pre-planned programming progression. Programming is thus presented as an activity that can be held even when many students are missing, a telltale tactic that both indicates what is prioritized and what is still respectable content for a math lesson.

Dual teaching is therefore tactic that accommodates the goals of the curriculum into the progression that the teachers want for their students. The reasons behind this choice come from students’ lack of programming knowledge as well as time considerations. Many teachers might be reluctant to change a well-functioning lesson structure from previous years if they do not expect it to result in significant benefits for their students. The teachers organize the
programming lessons so that they do not interfere too much with the rest of the math course. For example, teachers can choose to have some computer programming during the otherwise idle time after the national exams or dedicate half of a long lesson slot to programming “games”. The students normally know in advance when programming time will be allocated, and they will be asked to bring their computers or to leave their textbooks. The lessons devoted to computer programming are significantly distinct from the rest of the mathematics lessons, not only because of a shift in tools but also a shift in students’ attention. For example, whole-class interactive activities are replaced by written explanations and scaffolded exercises guiding the progress. A formal assessment of programming skills might occur in the form of assignments at the end of a programming unit but is rare.

When mathematics teachers who follow a dual teaching approach describe the way they include programming activities in their teaching, they are likely to point out the particular programming topic presented in each lesson and how it follows a designed progression path. Table 7.1 shows some examples of comments made during the interviews that endorse this approach.

Ariel shows awareness of the progress the students are making with computer programming and implements a pedagogical design path of increasing difficulty for the programming concepts that the students encounter. In Figure 7.1 Ariel one can see how the learning objectives and the success criteria of the lesson are specifically tailored to familiarize the students with programming processes and constructs. The mathematical content is present as the background topic on the average rate of change (Swedish: ändringskvoten) which is part of the syllabus for that particular course.

Dual teaching as a tactic gives Ariel the possibility to comply with the curriculum and at the same time distance themselves from the policy strategies by asserting their own teaching identity, confronting the way they will prefer to do things and the way they feel it is required from them, self-defined goals that justify chosen tactics amidst the fast implementation of the new curriculum.
## 7.2. DUAL TEACHING

Table 7.1: Excerpts of interviews revealing dual teaching.

<table>
<thead>
<tr>
<th>Teachers</th>
<th>Comments</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ariel</td>
<td><em>This [the rectangle area function] is more complicated, you see, because there are two [parameters], so that is the next step [...] so it is actually more like a formula than a function, but we call it a function because this is programming [...] This is, the way I see it, an example of ‘mathematical problem-solving with programming’ but it’s not necessarily the way I would choose to solve this problem.</em></td>
<td>A:1</td>
</tr>
<tr>
<td>Ariel</td>
<td><em>We did this exercise last time, when they [the students] already knew quite a lot of programming. [pointing to a worksheet about programming with loops to calculate interest rates]. In this lesson I want to introduce functions, you know, the concept of a procedure in programming (see Figure 7.1). I usually start with the factory metaphor, you put something in and something else comes out. They already recognize that from math.</em></td>
<td>A:2</td>
</tr>
<tr>
<td>Billie</td>
<td><em>So we would start with arithmetical operations, that’s one of the basics. Then I thought of these “guess a number” games, where you ask a friend to think of a number, and then perform certain operations on it and you can predict what the final number will be. This, you can program in the beginning, when you still do not know much Python and practice the arithmetic operations and how to assign values to variables. So I gave them one of those number games and the students would program it. Then the task was to see the pattern and come up with your own “Think of a number” game. Some students wanted to use random numbers, and then I had to bring it up for the whole class [the random python library] because this was new for most of them.</em></td>
<td>B:2</td>
</tr>
<tr>
<td>Chris</td>
<td><em>It was the third time we would have programming [December after National Exams]. I thought we would need some repetition of what we had done before. Programming lessons are so far apart! But loops were new. I thought, can we do it with some math from the course? And I thought of solving equations. Maybe a little odd. But I know that many [students] do not know what it means to solve an equation. They can do it, move x there and such, they know the process, they are so focused on the process itself that they lose the meaning. So using programming, they could test many different values. The limitations are obvious, but also worth discussing. We only tested integer solutions, positive numbers, then expanded to negative and then some even tried to use rational numbers, easy fractions, that way we also got to practice the number sets!</em></td>
<td>C:2</td>
</tr>
</tbody>
</table>
CHAPTER 7. TACTICAL RESPONSES

7.3 Interspersed Programming in mathematics

Another approach to align mathematics teaching to the curriculum revision is here called *Interspersed programming* (IP). Teachers who adopt this position teach the normal sequence of mathematics units and recur to programming in those situations in which programming would be a reasonable tool to use within the topic. This means attending mainly to one progression path, that of traditional mathematics, but gathering tools from different areas. The name *Interspersed programming* evokes the idea of dispersed, varied, and nonadjacent samples throughout the mathematics course. In these cases, programming is less prominent in the explicit unit plans, but implicit in the everyday practice and in the lesson development. The expectation is that students will understand the utility of coding in different mathematical situations by reflecting on computational patterns and taking advantage of powerful data structures.
7.3. INTERSPERSED PROGRAMMING

The occasions when programming is used are scattered along the course and are not necessarily evenly distributed. The teacher might prepare a programming activity in order to advance students’ comprehension or skill in the mathematics unit that is being studied or as a way to introduce a new topic. This relegates the programming being used—and learned—to the needs of the mathematics progression. Since programming is not taught explicitly, there is no preconceived progression in that respect. For that reason, the code that students encounter at the beginning of the mathematics course might be more complex and involve more advanced programming than code in later examples. Students engage with the programming activities according to their programming skills and interests, ranging from observation to tinkering to creating their own solutions, but the focus remains on learning the mathematical concepts and procedures. In the assessment situations coding is, at most, optional, i.e. some exercises can be solved with computer programming if the student chooses to do so but the solution is evaluated by the quality of the result rather than the method.

When teachers who resort to interspersed programming explain the way they plan their lessons, their starting point is the mathematics topic that needs to be covered at that time and how a computer program is intended to facilitate learning in this specific context. It can be to use computer power to elude tedious calculations, to achieve better precision, or to exemplify the process behind a formula (see Table 7.2).

Interspersed programming has the tactical undertones of a work-to-rule campaign, in which the prescriptions of the curriculum are met to the letter, but no extra effort is put into bringing students up to speed with regard to programming knowledge. This tactic allows teachers to include computer programming in a way that suits their pedagogical design for the mathematics content and opens the possibility for students to explore and practice with mathematical concepts in yet a new way with computer programming.
## Table 7.2: Excerpts of interviews revealing interspersed programming

<table>
<thead>
<tr>
<th>Teachers</th>
<th>Comments</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dominique</td>
<td>One shouldn’t force the programming in the course. It is important that programming feels relevant. And therefore, doing simulations when we are studying probability is a straightforward choice. They can be done in many different ways and it helps not only to solve the problem but also the understanding, for certain situations that are not so obvious right away. Take for example the Monty Hall problem. Or why the house wins in the long run on all casino games.</td>
<td>D:2</td>
</tr>
<tr>
<td>Evin</td>
<td>This problem is great, this is the second year I let my students work with it here [the unit plan about logarithms]. What is so good is that you take up logarithms, which otherwise is not the first thing you think of when it comes to programming, and that it can be adapted to how much programming the students know. Because that can be very different. So far, I am the one doing most of the programming anyway, but they get the file to tinker with. They work in pairs, so maybe someone who knows more programming can help the other.</td>
<td>E:2</td>
</tr>
<tr>
<td>Francis</td>
<td>Symmetric difference quotients and side measures, they are treated very superficially in the textbook. This is the foundation for the definition of derivatives, but it requires iterative numerical calculations, and it gets boring with the calculator if you are to do it for real. So here we coded. We built the program together. And then I took the opportunity and used a variant of Cobb-Douglas [production function] that they might benefit from if they study economics, these students are from the economics program […] They should minimize the cost of manufacturing a certain number of units, and what they can vary are the resources, the machines. […] But the function itself is blackboxed, I give the code for it and in there they should not change anything. Just adjust the precision they want.</td>
<td>F:3</td>
</tr>
<tr>
<td>Gaby</td>
<td>I had this same exercise in the second quiz [geometric sums], of course, they could have programmed it, we have done that in class, they had their computers and that, but none of them did. And that’s fine. I think there is a strong belief that an algebraic solution is worth more so the few students that would have known how to program, they also know that there is a formula.</td>
<td>G:3</td>
</tr>
</tbody>
</table>
### 7.3. INTERSPERSED PROGRAMMING

<table>
<thead>
<tr>
<th>Teachers</th>
<th>Comments</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaby</td>
<td><em>But it’s not that damn difficult [programming]. At this level you only need the really basic: while-loops, if-statements, that’s it. And they should have seen that in primary school. My goal is for them to learn to see when programming makes sense in mathematics. When it can be a good complement. It’s another tool, like the calculator, more powerful, or like GeoGebra [interactive math environment with dynamic geometry]. It’s used when it’s needed. So, I show problems where programming is good. No point in making a program to calculate one rectangle’s area, but very much so if you need to calculate a thousand areas, like you do for a Riemann integral. Then you sometimes have to go outside the textbook, because there, all tasks can be solved without programming. But then we end up with the problem we discussed before [relevance and assessment in national exams]</em></td>
<td>G:4</td>
</tr>
<tr>
<td>Hanne</td>
<td><em>So I do not assume that they can program, but I do not teach programming as I do when I teach programming courses. I teach math. And then I show when programming can be useful. But then it will be mostly examples that I draw. Like this cool prime-number sieve [...] There is no time to have proper computer programming if we have to start from scratch. But for those who can, they understand what the program does. The others, I don’t know, they get to see the code grow and I explain, so they have seen that you can program in math.</em></td>
<td>H:4</td>
</tr>
<tr>
<td>Inge</td>
<td><em>The Monte Carlo method to calculate integrals gives a very different view on integration, now it has to do with probability! And coding it raises the solution above all the magic of antiderivatives. There may not even exist like in this example [There is no analytic primitive function of (e^{x^2})]. It’s actually quite interesting for those students who like hands-on, but you still need to do a bit of math, you need to have an idea of the largest value to select the right interval [...] A student tried to do it with volumes instead of areas. But it was too much, and we ended up discussing why inside/outside a body is a much harder issue than above/under a line.</em></td>
<td>I:2</td>
</tr>
</tbody>
</table>

When Gaby mentions that they need to go outside the textbook (G:4) they give a glaring at what they consider the core of the course and the way they choose a deviation from a path that was officially recommended. This is a form of resistance that may manifest in less obvious practices of dissent, an oppositional tactic disguised as obedience.

The activity that Inge proposes introduced their students to the Monte Carlo method to approximate the definite integral of a function (Inge:2 and I:3).
CHAPTER 7. TACTICAL RESPONSES

It takes quite a lot of thinking, to figure out which area you need to sample. So, when the program returned the wrong number, they needed to go back and reflect on the conditions. Which fraction of the rectangle are we hitting? [Monte Carlo simulation had been explained to the students with the metaphor of throwing darts.] Some functions had a negative domain, like this one [points at $f(x) = \sqrt{1 - x^4}$], or negative range or the extremes were not at the ends of the interval, so, yes, it takes some calculations before you can plug in the right numbers in the program [...] We’ve been doing a lot of function analysis this semester, and it came handy. I think we got to deepen many of the concepts that are relevant in the course.

![Figure 7.2](image)

Figure 7.2: Inge’s code example using Monte Carlo simulation to find the area under the graph of the function $f(x) = e^{x^2}$

The students are presented with a program that solves the problem for a given function and they are encouraged to tinker and tamper with it to get a
7.4. COMPARATIVE ANALYSIS

grasp of which factors may affect the accuracy of the result (see Figure 7.2).
Several functions and different intervals are treated to try to build a robust
program that can handle variations. During the interview, the mathematics
topic of the lesson—numerical integration—is thoroughly explained, relating
it to other topics such as derivatives to find the extreme values and probabil-
ity to understand the uniform distribution. It is even extended to advanced
topological concepts outside the syllabus. When asked about which skills the
students were expected to develop with the activity, mathematical conceptu-
alizations and reasoning about proportions were mentioned whereas nothing
was said about programming dexterity. This omission may be attributed to
the context of the conversation—learning activities with programming. The
teacher might have considered that the development of programming skills
could be taken for granted. It might also be the case that the activity itself was
not designed towards any particular programming development.

7.4 Comparative Analysis

When the unit plans and the observation notes were analyzed in the light of
dual teaching and interspersed programming, new insights developed and
delineated the description of these approaches as tactics. Contrasting the con-
tents in the interviews with the teacher’s unit plans further nuanced the tacti-
cal divides with both intentions and expected outcomes. For example, Chris
had planned for a “math quiz on chapters 2:3-2:6 and 3:1-3:3 on Wednesday”,
specifying that no digital tools were allowed and that the following lesson
after the quiz was to be devoted to programming (for-loops), before starting
with a new unit. This supports the idea of a Dual Teaching tactic, in which
programming is kept as a separate knowledge matter, secluded from mathe-
matics.

The differences between the two tactics, Dual teaching and Interspersed
programming, reach beyond the mere separation of the subjects in terms of
tools and time or whether one or two progression lines are implemented (Ta-
ble 7.3). The tactics are unveiled also in the different ways that classroom
activities with programming are conducted and in the methods that teachers
use to appraise students’ understanding. Fundamentally, there are different
goals regarding what is to be learned and how programming is understood.
Teachers who choose to intersperse programming demonstrations within regular mathematics lessons embrace the instrumental capabilities of code. Those teachers who instead dedicate programming a distinct —dual— treatment, relate also to other aspects of programming, such as methods for debugging code or reusing solutions in different contexts.

**Table 7.3:** Comparative between Dual teaching and Interspersed programming

<table>
<thead>
<tr>
<th></th>
<th>Dual Teaching</th>
<th>Interspersed programming</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Progression</strong></td>
<td>Two lines, one for mathematics and one for programming</td>
<td>One progression line only, for mathematics</td>
</tr>
<tr>
<td><strong>Teaching Goals</strong></td>
<td>Advancing mathematics comprehension and learning programming skills</td>
<td>Advancing and extending mathematics comprehension (with the use of programming).</td>
</tr>
<tr>
<td><strong>Schedule</strong></td>
<td>Explicit schedule for computer programming activities that does not intrude much on the normal flow of mathematics lessons.</td>
<td>Ingrained and scattered throughout several lessons, not always explicit in the topic plan.</td>
</tr>
<tr>
<td><strong>Student activity</strong></td>
<td>Programming activities follow scaffolding learning that is gradually withdrawn. Code is sometimes provided to tinker with and sometimes co-generated</td>
<td>Programming activities are teacher-centered. Students’ participation is expected in reflections and analyses. Code production is not expected.</td>
</tr>
<tr>
<td><strong>Assessment</strong></td>
<td>The programming activities are to be completed during the assigned lesson time. Assessment occurs in parallel as the teacher monitors the progress</td>
<td>Programming is in some cases possible to use to solve exam problems, but it is never required.</td>
</tr>
</tbody>
</table>
Chapter 8

Tactics that become Practice

The first five years after the introduction of programming in the mathematics curriculum have brought along changes in teaching that are starting to show signs of stabilizing in sustainable practice. This chapter is an attempt to outline what might be left after the first implementation efforts lose their spark of novelty. The official strategies and the induced tactics that had been explored in the previous chapters are both events in a fluctuating historical context. Practice too will fluctuate, as it has invariably done before, but it will do so less piercingly and with more consistency.

The distinction here is indeed a question of time scales, like that between waves and tides. The analysis of the data will therefore be guided by the continuity aspects revealed in the interviews. Teachers’ initial resolutions, trials, and experiments with programming in mathematics are sometimes reinforced by means of perseverance and the teachers’ mature reflections on their past experiences. Other tactics need to be refined or updated and yet some others are discarded.

The following sections present the results of the second iteration of interviews with respect to the choices the teachers make when including programming in their mathematics lessons compared to those they made earlier. The relevant categories that emerged along this continuum (kept-modified-obsolete) are here explained and illustrated with excerpts from the interviews and unit plans. These categories highlight the principal characteristics of the programming activities that were in use after the teachers had been imple-
menting programming for some time and serve to structure the subsequent sections.

8.1 Clear learning outcomes

One of the main factors that directed the continuation of activities in the second iteration was the clarity of learning goals. The teachers acknowledge having gained a better understanding of the learning outcomes they hope to achieve through programming activities. They were able to articulate how some programming activities supported the development of students’ mathematical understanding or their programming skills and how others were hindering or confounding. Evidence of this sort of reflection is found in several passages.

For example, Chris justifies the chosen programming activities by motivating their explicit learning objectives, in this case, learning about conditional branches in Python (C:3).

The first years, I think I mainly transferred my programming course into the mathematics slots devoted to programming. I already had a working structure ready to use. But I have polished it, because I want the students to achieve a bit different goals. Actually, I have stripped it down to the essentials. So if I want the students to learn about conditionals I give a MWE [Minimal Working Example, programming slang]. For that Python provides unparalleled simplicity, really […] Still I keep the order that I think is best for learning programming (conditionals before loops, for example), but I do not strive for the most effective solution or the most streamlined code—which is also something very pythonic— but the most intuitive.

Clarity is posed as the main reason behind the example in the programming activity in mathematics. The example that Chris shows in this respect is a basic conditional with no frills, answering the question of whether one is allowed to obtain a Learner’s Driver Permit depending on one’s age (Figure 8.1). The given program is to serve as a base for the code to be extended with further age restrictions for other licenses, such as taxi driver or heavy lorry.
When Chris explains the exercise, they mention choosing minimal working example (MWE) when programming in mathematics. They mean that this is not necessarily the same as the neatest way of writing Python and compare the snippet with the equivalent choice for the students in the programming course, which takes just one line rather than four (handwritten note in Figure 8.1). This highlights the different approaches used to cater to the needs of mathematics students learning programming and those studying programming as an elective subject. Programming in mathematics gives just the essential rudiments to solve problems while the students in the programming course are expected to utilize the technical advantages of Python regarding conciseness and effectiveness.

Figure 8.1: Two different solutions to the same problem addressing the needs of students in two different subjects (elective programming and mathematics) as explained in C:3 (2022).

The difference reveals two distinct approaches to learning programming with two different goals. While Chris’ practice still matches Dual Teaching tactics, treating programming as a ‘subsubject’ within mathematics does not fully correspond to teaching a separate programming course. The learning encouraged in Dual Teaching orients programming towards problem-solving, logical thinking, and algorithmic reasoning and is less concerned with the affordances and technicalities of the specific programming language. This de-
liberate distinction allows Chris to advance programming in mathematics in
a much tighter schedule as well as to offer a clear vision of the expected learn-
ing outcomes. The “Theme: Conditional Statements” (Swedish Tema: Villkos-
ratsar Figure 8.1) is introduced with only two subtopics: True/False expres-
sions (i.e. boolean expressions) and the if-else construct. Here too, a minimal-
ist approach is the preferred design to convey the contents.

Similar insights can also be found in the sustained practices of teachers
who opt for Interspersed Programming. In these cases, the clarity of the learn-
ing outcomes relates largely to mathematics knowledge rather than program-
ing. During the first iteration, Inge had shown an exercise in which pro-
gramming had pushed the lesson towards advanced mathematics topics. “A
student tried to do it with volumes instead of areas. But it was too much, and we ended
up discussing why inside/outside a body is a much harder issue than above/under a
line (I:2)”.

Back then, Inge pointed out that this discussion might have been for the
benefit of only some students. In contrast, instead of deploying the full gamut
of possibilities, Inge now explains how reining in the versatility of their pro-
gramming examples and concentrating on fewer selected learning goals has
helped them to achieve more accessible lessons (I:4). They pointed out how
they were able to keep the students motivated with fewer examples and less
sophisticated programs which also contributed to a more balanced use of tools
in mathematics.

The program with the integral... [finding the definite integral of
a given function over a specified interval using the midpoint
rule] you know, there are so many variations and improvements to
include there. Like, letting the user input the interval, or the number
of rectangles, or even which function. And, is the midpoint the only
possibility?... there is a learning opportunity for each decision but
together they make the program unattainable. So one of the lessons
—a lesson for me that is— is that you don’t need to do it just because
you can, better resist the temptation to ... show off the power of
programming so to speak.

Here Inge reflects on the broad range of capabilities and options at hand.
These are options offered by the mathematical topic itself that, because of the
laborious calculations they entail, are seldom fully inspected in the textbooks.
Programming would allow for a more varied and interactive exploration of numeric integration but pursuing this, Inge concludes, would counteract the simplicity that they intend to convey. Reducing unnecessary features allows Inge to bring forward the essence of the integral calculations. The trade-off here is not that of programming versus mathematics but of keeping the whole activity moderated, which means narrowing the scope of the mathematical concept and narrowing the scope of programming functionally to gain depth in learning.

Despite belonging to consecutive mathematics courses, the programming aspects of the activities mentioned in I:2 and I:4, reveal only negligible differences regarding the difficulty in the code —both need external libraries, user-defined functions, and loops. This, along with the purposeful priorities on content, aligns with the philosophy of *Interspersed programming* in that it is the utility that justifies the use of programming.

Another passage that highlights the importance of reflecting and understanding the learning outcomes of the chosen programming activities comes from Evin. This was apparent when they revised the exercise with logarithms that was part of their unit plan in 2020 (E:2 and Figure 6.6). During the second interview, Evin elaborates on why it is important to align the teaching activity with the desired learning and reflects on the risk that programming might blur rather than assist comprehension.

*That one! I used it to teach a lot of mathematical concepts with programming, but now I’ve got a better grasp of what works. The concept needs to be in focus, but also the utility, so no, not that way, I mean, [the code] is obscure even to me now! Logarithms by trial and error is not the way they are used, [...] they [the students] got confused, thought that logarithms had to do with very small intervals. Instead, I have now this one [Figure 8.2] on plotting sound intensity, to see that logarithms are about managing small numbers AND large numbers at the same time. [...] It looks like a completely different activity, and I think the students see it completely differently now. But, well, the programming part is still about getting some code and trying to understand what is happening there, to make the necessary changes.*
Evin’s reflection in E:3 is an example of how the tinkering approach has remained at the core of a whole new activity with logarithms. The changes pertain to the mathematics domain and are explicitly designed to convey clarity and practical use. A possible misunderstanding with respect to the concept of logarithms was narrowed down to the previous programming activity and informed the changes needed to enhance comprehension. The revised activity now connects logarithms with real-world applications in physics, a subject that Evin also teaches to the same group. By using the synergy between mathematics and physics, they aim to deepen students’ understanding of logarithms and the practical applications of logarithmic scales in scientific contexts.

From the programming perspective, the new activity might be more engaging because of the visualizations, but not necessarily more difficult. Some new constructs have been added, such as functions, lists, and a graphing library (Figure 8.2). On the other hand, the new program is now sequential, significantly alleviating the cognitive burden previously imposed by the while-loop and the conditional branch. The changes seem to be independent of any predefined progression in programming but rather tailored to suit the needs of the mathematics topic, as was characteristic of Interspersed Programming tactics.

By focusing on the expected learning outcomes, the programming activities that remain in the course have been consciously abridged to bring forward coherent learning aligned with the topic, whether it is in mathematics or programming. The teachers have learned from previous attempts how to avoid too cluttered or overloaded exercises that can be the result of overambitious expectations.

8.2 The faces of programming in mathematics

The three examples above (Figures 8.2 and 8.1 and passage I:4) evince another important aspect of the long-term teaching practices: the coding activity as such.

Coding activities in the classroom can be instantiated in many different tasks. Section 3.1 showed how the programming teachers’ toolbox encompassed a range of approaches common in programming instruction. The dif-
8.2. PROGRAMMING TASKS IN MATHEMATICS

Figure 8.2: Evin’s Python activity using log_{10} to compress a wide range of values into a more compact and easily interpretable scale (2022).

Different types of activities covered there presented increasing levels of freedom, creativity, and decision-making required from the students. These degrees of autonomy are also a relevant factor when the teachers choose how to orchestrate the inclusion of programming in their mathematics lessons. However, the actual examples seem to be confined to a slender range of variations in which time emerges as a distinct constraint.

In the first iteration of this study, we found several examples of demonstrations (C:1, H:4 and I:2), tinkering (e.g. E:2 and F:3) and scaffolding (e.g C:2). Francis mentioned an activity in which the students and the teacher developed a program together (F:3) and Billie’s students built a program from the
CHAPTER 8. TACTICS BECOME PRACTICE

ground up following a template (B:2). The second iteration of interviews did not widen this catalog. Rather, the remaining activities showed an inclination towards further boundaries to student participation and more input from the teacher.

Demonstrations

Demonstrations were ubiquitous in both iterations. In excerpt I:4 Inge refers to a programming example illustrating a mathematical procedure. It shows also an undisguised tactic of harnessing, holding back the power of Python, showing without showing off. Using demonstrations was therefore Inge’s way of controlling the boundaries of the activity to center the learning outcomes.

Also, time needs to be bridled in a classroom setting, as Hanne points out:

\[
\text{I hope my students at least have an understanding of which kind of problems are suitable to program […] another factor there is to know how to represent the information so that the computer understands. I try to show examples of different data structures, not only lists, like finding the greatest common divisor of a range of numbers using binary trees […] of course, that means quite advanced programming, and I just showed the algorithm, not the code, that would have taken the whole lesson.}
\]

H:5

In the example above (H:5), Hanne’s students get to see how the code behaves in real-time and what kind of problems can be solved with programming. They underscore the issue of data representability as a premise for finding a programming solution. While data structures and algorithms are two interrelated concepts, the mathematics problems that the teachers provide tend to focus mainly on the process. The exercises often rely on given numerical types (integers, decimals, etc) that already exist in the programming language, with no further need to build up complex structures. By adding different ways of representing data, Hanne balances those two aspects of computational models.

Hanne’s algorithm demonstration is not unique in concealing the code. In F:1, Francis engaged with programming in mathematics by only sharing with the students a program’s interface, that is, not the code itself but the execution
8.2. PROGRAMMING TASKS IN MATHEMATICS

of the program. “So I made the program that does just that, and we tested it” [F:1]. This kind of demonstration, where the code is hidden from the students, has no direct equivalent in traditional programming education but seems to fulfill a particular purpose in mathematics: to underscore the connection between mathematics and computation.

The teachers motivate their choice of demonstrations with similar arguments as those presented in the literature (see subsection 3.1.1). They want their students to gain firsthand insight into the dynamic behavior of the code and the problem-solving potential inherent in programming. Demonstrations can be accommodated into smaller lesson segments and demand less preparation from the students which seems to fit well with Interspersed Programming. They can even allow for spontaneity as in C:1, where a more interactive activity would have required preparations from both teachers and students.

Tinkering

Different variations of tinkering activities are also among those practices that abide the test of time (subsection 3.1.2). Both Chris and Evin show examples in which the students are provided with code snippets that need to be altered to achieve new functionality (C:3 and E:3). Also here the tinkering activity has a slightly different framing than what is common in programming education, where code is often provided to learn how to find and correct errors (debugging) or to prompt reusability. Instead, tinkering with code in mathematics is instantiated to target conceptual units (boolean operations in C:3 and logarithms in E:3) and to relieve the students from the time-consuming and error-prone task of writing whole programs from scratch.

On the other end of the spectrum, Gaby advocates for the significance of algorithmic thinking: “Designing an algorithm that solves a problem will always have more value than executing the steps of a process that someone else has come up with (G:2)”. Designing algorithms is at the core of programming education and one of the most challenging endeavors in Computer Science (e.g. Krishnamurthi and Fisler, 2019). It belongs in the realm of problem-solving and requires a good understanding of the affordances of computation as well as the constraints of the problem. In the first interview, Gaby argued that “I think there is a strong belief that an algebraic solution is worth more” (G:3) to justify why students might have opted out of coding a solution in a test. The
impersonal expletive pronoun *there* subtly implied that Gaby’s personal opinion could have differed from that belief. Nevertheless, none of the teachers provided examples of activities where the students themselves were asked to devise and program an algorithm to solve a problem, which could be a sign of the difficulty of such exercises.

### 8.3 Connections to real world mathematics

A third characteristic of the kind of activities that the teachers kept in their repertoire is the connection to application-oriented mathematics and their relevance outside instructional contexts. Several of the examples from the second iteration manifest this perspective. These ideas can be gathered from fragments such Evin’s “*Logarithms by trial and error is not the way they are used*” (E:3) or in the following reflection from Hanne:

> I want my students to feel that mathematics is for real, not just the subject that one learned in school and never use again, so I want to find real examples from their day to day life. We have simulated how a virus spreads in a community […]. They can relate to that, like “What if the virus only catches in 25% of encounters?”, and programming is great for that kind of simulations […] And because you are in charge of the parameters, you trust the results.

Simulations of real-world phenomena come from many different areas such as physics experiments, population dynamics, or financial models (e.g. F:3 and A:4). Classic mathematics is also a source of real-world matter. The recurring concept of probability is instantiated with programming examples in both Chris and Dominique’s activities (C:1 and D:2). These simulations enabled students to observe and control the behavior of practical entities in action. Despite modeling and simulations no longer being expressly included in the second revision of the curriculum, the teachers continue to choose these kind of entry points for programming in mathematics. In G:2 Gaby says “*it doesn’t really matter how much of it is explicitly stated in the course syllabus or subject syllabus, you just use it when it fits*”. This fragment is later accompanied by the following argument:
8.4 From tailored activities to equity

Mathematics is much like craftwork, like learning to paint portraits, we are not seeking a fast representation, that we can get from a photograph. We are seeking an understanding and a skill, so you cannot take shortcuts, and programming is sometimes a shortcut. But if you are constructing an algorithm to solve a real-problem, then the craftwork is there, you are putting in the work, the creativity, there is learning happening.

This reflection adds dedication and attention to detail to the list of motives that the teachers give for using real-world examples when programming in mathematics. The statement cuts both ways; it disapproves of programming replacing learning mathematical procedures but commends its applications for solving new problems.

8.4 From tailored activities to equity

An interesting difference in the second iteration was the teachers’ views on adaptability. During the first years, several teachers prepared activities that could be tailored to the different levels of programming knowledge that they were expecting from their students. It could be for example providing tinkering exercises with different degrees of difficulty such as Evin’s activity with logarithms: “What is so good is that you take up logarithms […] and that it can be adapted to how much programming the students know” (E:2) or simply demonstrations that had different connotations depending on the student’s prior knowledge, as alluded by Hanne in “But for those who can, they understand what the program does. The others, I don’t know, they get to see the code grow and I explain” (H:4). Inge too recognized the different preferences among their students when they mentioned that “It’s actually quite interesting for those students who like hands-on” (I:2).

In the second iteration of the interviews, there were no such mentions of exercises catering to varying levels of proficiency. The programming activities were curated to be accessible to all students. This is in part related to the clarity of learning goals examined in section 8.1 but also explicitly mentioned as a rationale when designing unit plans.
I make these lists with concepts, one for every topic, so I made one for programming too. It works as a checklist when preparing for the exam; do I know what an isosceles triangle is? do I know what an inequation is? And so on, and I realized that some words have a slightly different meaning in the context of programming, like a variable, or the equal sign, it’s confusing and the glossaries are a way of making it clearer, I hope. [...] because if you have programmed before, you know the meaning in the programming context, but most don’t. I think the glossaries help to get everybody on board. If I put some effort into something, it means that it’s important, that it isn’t obvious, nobody needs to feel foolish because they mix up concepts.

Striving for a larger degree of uniformity aligns with the trend of programming taking up less time than it did at the beginning. This approach promotes equity and facilitates an inclusive learning experience for all students, regardless of their programming background.

[I teach] definitely less programming now, but maybe more effective lessons. We get on with the programming faster, less downtime with servers acting up and the exercises are shorter. And a better way of sharing the code templates. In physics, there is this POEE model [Predict-Observe-Explain-Explore] and that works for programming too. I show a bit of code, they [the students] guess what it does, they run it to see what it actually does, and well, instead of explaining, they might need to add some comments to the code. Now they can use the commands themselves. Very simple exercises, very low threshold.

Billie prepares less demanding and well-orchestrated activities to help create a supportive classroom environment where their pupils follow a predetermined path. They describe how the activities are carefully scaffolded from a template and also how code comments are used to secure understanding. The comments in this case fulfill the role of labels for sub-goals that was common in traditional programming education (section 3.1.2).

Inclusiveness is also Evin’s intention with the activity in Figure 6.6, where the students needed to use a given Python program to plot the relative loud-
ness for a series of sounds. The worksheet already provides all the necessary code and a table with the acoustic energies of eight common sounds, which confines the range of acceptable answers. In the following passage (E:4), Evin reflects on the drawbacks of open-ended exercises.

This one leaves little room for interpretation, but also little room for errors. The risk with open-ended exercises, . . . I may end up spending most of the lesson helping the students who make the most advanced programs, because they somehow, they manage to get into the most twisted errors. This one works for all.

Evin’s unit plan does not include programming lessons as such but their students have the opportunity to program in one of the end-of-course assignments if they choose to do so. Open-ended activities, in this context, hold great potential for advanced students who have already acquired a solid foundation in programming. However, when time is scarce, opting for a more structured programming activity is a harnessing tactic that ensures that more students can engage and succeed.

Awareness of equity emerges as a central issue of the sustained practice of programming in mathematics. After the initial trials, the teachers seem to settle for simpler activities that reach the whole class avoiding intricate programming tasks. Fairness is also important in relation to assessment (e.g. H:8) which will be analyzed in the next section.

8.5 Assessment and standardization

Two factors remained a significant obstacle hampering the implementation of programming in mathematics: assessment and standardization. During the first years of implementation, several teachers pointed out that it would take some time for the teachers and pupils to catch up with the new requirements. An obstacle in that direction was the lack of official directives regarding programming languages and proficiency levels and how the new requirements would be reflected in the National Exams (e.g. F:2, A:3, G:4). Comments about programming languages were less prominent in the second iteration but the criticism against fuzzy guidelines persisted (e.g. C:4).
We [the teacher and their students] had this discussion about what programming experience they had, and programming languages came up. I tried to make a point that most programming languages are equivalent, and I showed how conditionals worked in Excel. [...] Still, I think it would have been better if there had been an official decision on a programming language like we have an agreement on which mathematics symbols we use.

The phrasing “an official decision” (C:4) indicates that there might already exist an informal understanding regarding which programming language is generally taught in compulsory school. Nevertheless, the National Exams were expected to inform about appropriate praxis and even motivate a faster implementation. During the second iteration, it became apparent that the teachers’ expectations had not been fulfilled. In I:1, Inge remarks that “After 5 years programming has not appeared in the National Exams, not once” which suggests that it would have made a difference for their teaching practice. Also, Hanne insinuates the standardizing function of the National Exams but relishing instead on the independence that their absence granted “there is no National Exam for that course, so more freedom in how to test” (H:7). This fragment alludes to the closed nature of tests in courses where the students’ final grades are dependent on their performance in the National Exams. The influence of assessment is a recurrent commentary:

I see that first-year students are not as scared as they were four-five years ago, and not as impressed either [by programming]. They have seen code, I think all of them have, but most of them cannot program yet. And they have probably heard that one can pass the courses without it.

While final assessments certainly entail incentive value, Hanne directs the attention toward the broader implications of learning programming. Students need to be prepared for future endeavors beyond the scope of current mathematics courses.
Of course the final tests are important, but there are many other things that can boost motivation. These students, most likely will need to program in higher education, whether they study engineering or science or something else, they need to be prepared for that too. They don’t want to lag behind, particularly if they plan to study abroad, their peers there have surely learned to program in high school.

Assessment is carefully organized in mathematics teaching practice. Tests, quizzes, and exams are explicitly stated in every unit plan and are often preceded by lessons dedicated to revision (see unit plan in Figure 5.2). These are time-consuming duties that teachers and students honor and uphold. Programming in mathematics seems to keep escaping these examination practices and was not included by any of the teachers other than to survey students’ prior knowledge at the beginning of the course. One possible reason is given in the following argument (H:8).

I’ve seen math tests with programming exercises, but I’m unconvinced. It was something like “Why does this program crash?” and it was some division by zero. We don’t have stipulated programming content in Sweden so it wouldn’t be fair, and if they don’t know how to solve the problem, is it because they don’t know why rational expressions are tricky [when the denominator is zero] or is it because they don’t understand python? What am I really testing?

Hanne brings forward the impartiality expected from assessment practices and offers an explanation that accounts for the conflicting objectives of programming in mathematics: the idea of programming as a tool rather than as content. This line of thought would place programming among the possibilities available for the students to solve a given problem, where the teacher should appraise whether the solution fulfills the requirements, regardless of the neatness of the code.

8.6 Nuances in Dual and Interspersed tactics

In the survey part of the second round of interviews, the teachers were asked about the courses they taught with programming and how the current pro-
programming activities there compared to the previous ones in terms of time and difficulty. After being introduced to the ideas behind Dual Teaching and Interspersed Programming the teachers were also invited to assess their current integration tactics. Their perceptions on the matter are reported in Table 8.1.

The two tactics discerned in the first iteration were still in use and the majority had persevered with more or less the same approach as they had done before. There were nevertheless some new insights and discrepancies that contributed to a more nuanced understanding of the dichotomy.

One of the teachers had partially migrated from Interspersed Programming to Dual Teaching with their new students but continued with interspersed programming in more advanced courses. This choice was justified by the pressure from peer teachers and students. Inge explains:

*I had all these ideas, we can program this, and we can program that, and I kind of expected that my colleagues would step up and follow suit. But in the end, it was tiresome, and I felt singled out, and now I just do more like the rest of them, we teach a little programming once in the first course, not even we, they have these people coming over from the university, and the pupils are gathered in the assembly hall with their laptops and do half day python. [...] I think this came about because many [teachers] were complaining that they didn’t get paid time off for learning programming, and maybe this solution was cheaper, I don’t know.*

Ariel argued that other teachers if not themself would hopefully reap the fruit of their dual approach later.

*This is during a transition period, until you get in students who can program and then in later years, of course, or maybe in Physics, but maybe you shouldn’t throw in advanced code if the students can’t program at all*

Their motives resemble those of Dominique when they said that the students who were about to graduate would not receive any programming instruction. “It wasn’t worth it to start changing. But next time, maybe, in two years, then the threshold won’t be so high” (D:1).
It seems possible to find intermediate tactics between *Dual Teaching* and *Interspersed Programming*. In connection to their previous answers, Evin concedes that they might have a tendency towards interspersion but adds

*The part that the students do [now] is easier, they don’t need to understand every single line, but it is not like I don’t explain anything. I mention the variables, and the exponential notation, which resembles the T84® [calculator] that they use in physics. I wouldn’t say that I am unaware of their previous knowledge. I still want them to complete the worksheet and learn something from it.*

This reaction shows that *Interspersed Programming* can benefit from collateral attention to the progression of programming knowledge even when programming is used as a tool to learn mathematics. The teacher makes use of parallelisms with other tools and builds upon programming ideas that have been previously used in the classroom.

**Table 8.1:** Evolution of tactical approaches to teaching mathematics with programming

<table>
<thead>
<tr>
<th></th>
<th>Main tactic first iteration</th>
<th>Main tactic second iteration</th>
<th>Changes in programming time</th>
<th>Changes in programming difficulty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ariel</td>
<td>Dual</td>
<td>Dual</td>
<td>less</td>
<td>easier</td>
</tr>
<tr>
<td>Billie</td>
<td>Dual</td>
<td>Dual</td>
<td>less</td>
<td>same</td>
</tr>
<tr>
<td>Chris</td>
<td>Dual</td>
<td>Dual</td>
<td>same</td>
<td>same</td>
</tr>
<tr>
<td>Dominique</td>
<td>Interspersed</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Evin</td>
<td>Interspersed</td>
<td>Interspersed</td>
<td>more</td>
<td>easier</td>
</tr>
<tr>
<td>Francis</td>
<td>Interspersed</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Gaby</td>
<td>Interspersed</td>
<td>Interspersed</td>
<td>same</td>
<td>same</td>
</tr>
<tr>
<td>Hanne</td>
<td>Interspersed</td>
<td>Interspersed</td>
<td>same</td>
<td>same</td>
</tr>
<tr>
<td>Inge</td>
<td>Interspersed</td>
<td>Dual/Interspersed</td>
<td>less</td>
<td>easier</td>
</tr>
</tbody>
</table>
CHAPTER 8. TACTICS BECOME PRACTICE

The three chapters in part III shed light on the strategies, tactics, and long-term outcomes associated with the introduction of programming in mathematics teaching.

Chapter 6 provided an overview of the strategic onset, focusing on the changes in the curricula for both the compulsory school system and non-compulsory education, the programming activities that were developed to facilitate the transition, and the influence of the curriculum latent on the National Exams. After providing evidence of teachers’ awareness of the strategies, chapter 7 focused on the tactical responses for teaching mathematics with programming and two subsequent approaches: dual teaching and interspersed programming.

The present chapter has delved into the establishment of practice in teaching mathematics with programming. This included selecting programming activities with clear learning outcomes, keeping simple modes of programming instruction such as demonstrations, connecting to real-world mathematics, and considering equity and efficiency aspects.

In the next part of this work, the findings presented in these three chapters will be critically analyzed and interpreted, examining the consequences for mathematics teaching. Through a comprehensive examination of the research results, the discussion will convey the implications of these strategies and tactics for mathematics education and how they align with broader educational goals. Finally, and hoping to shed light on the complexities and challenges of this curricular transition, important insights for future educational reforms will be expounded.
Part IV
Chapter 9

The strategic vision and development of the curricular reform

The present thesis strives to provide insight into the practices of mathematics teachers when programming becomes an integral part of the curriculum that regulates the contents and methods that need to be taught in their subject. To delve into the meaning and relevance of the results, this fourth part of the thesis will explain and evaluate the findings with respect to the research questions, acknowledging the limitations that constrain their validity and offering the author’s personal interpretations and foreseen implications. In doing so, the significance of de Certeau’s understanding of everyday practices in terms of strategies and tactics is elevated to a working theory above the particularities of programming in mathematics and into the more general realm of institutional change.

The discussion is organized into three chapters addressing the three focus areas: the strategies behind the reform, the implementation of programming in mathematics, and finally reflections and lessons learned towards sustained practice.

The first point of analysis and discussion comes therefore from the strategies that define the framework of the reform in an attempt to address the ques-
9.1 Transverse classification of strategies

The strategies in section 6.1 were presented following the visibility dimension that is essential for this modality of action and that distinguishes it from the inconspicuous tactical operations. While the theory of everyday practices provides a useful classification of the activities into strategies and tactics, it delves mainly into the types of operations that form the tactics (subsection 4.1.4). For the strategies, there is another possible organization that rises from the perspective of functionality rather than visibility and power. This analysis traverses the previous hierarchical division and provides a new approach to the dynamic relation between tactics and strategies and their implications for practice. Strategies thus invoke and actualize a schematic and stratified ordering of social reality.

Data from the strategic policy documents and the interviews with representatives from the academies developing the National Exams are the main source of information for the analysis of the strategies. Together with the impressions of the respondents, these data were inductively examined in search of a structure that resembled the modalities of action (subsection 4.1.4). This analysis yielded seven categories and three meta-category clusters (Regulative, Facilitative, and Mitigative) that will be explained in the following sections. Figure 9.1 schematizes this structure distinguishing between actual strategies (dark font) and hypothetical strategies (lighter cursive font).

9.2 Regulative strategies

Regulative strategies involve creating laws, regulations, or rules to ensure that certain behaviors or activities take place or do not take place. de Certeau mentions regulative mechanisms as “a quantifiable and regulatable space” (de Certeau, 1984, p. 158) and “a law that legitimizes as ‘literal’ the interpretation given by socially authorized professionals and intellectuals” (ibid., p. 171, quotation marks in the original). The curricular revisions are the leading representation of regulative strategies underpinning the inclusion of programming in the mathematics curriculum. Both their inscriptive domain and
9.2. REGULATIVE STRATEGIES

Types of Operations

Regulative

Laws (Curricula)
Directives and Ordinances (Syllabi)

Facilitative

Public awareness
Professional training

Mitigative

Averting (fuzzy requirements)

Adaptive

Addition (National Tests)
Extraction (Fewer courses)

Figure 9.1: Scheme of types of operations in strategic modalities of action. The lighter cursive font is used to indicate those strategies that were mentioned in the interviews but have not (yet) been realized.

their recognition were established in section 6.2.

The regulative strategies are also the principal target of controversy and overt critique, especially after the first revision, and it is here we find the growing ground of many of the tactics deployed during implementation. One of the regulative strategies was to enforce the new curriculum simultaneously at all age levels from ISCED 1 to ISCED 3. This led to tactical responses such as designing for adaptability (bricolage) in Evin’s case (E:2) or delaying the introduction of programming in some courses to future cohorts that the teacher expected to be better prepared such as Dominique chose to do (D:1, la perruque).

In Table 6.1 the timeline for the regulatory decisions concerning programming in ISCED 3 was summarized. It shows that the regulatory documents have undergone frequent and swift changes, with five revisions taking place in under a decade. Particularly the second revision (Skolverket, 2022c) might have undermined the vision of a well-planned and thought-out reform. Teachers faced a sudden increase in their workload due to the need to understand and implement the changes in their lesson plans. Rapid and frequent changes added to the sense of uncertainty and stress exacerbated during the pandemic. The immediate consequences of these revisions could be seen in the tactics employed to keep up with evolving requirements. The teachers resorted to
la perruque, and harnessing tactics, in some cases dismissing the changes (e.g. H:2) and in some other cases resigning to a bare minimum that would tick off the necessary checkboxes (e.g. I:1 and C:3).

A third regulatory decision that affected teachers’ attitudes toward the reform was to leave programming as an elective subject available for all students and at the same time make it compulsory in some mathematics courses. In A:3, Ariel resents this arrangement and less explicitly so does Chris (C:3).

Dual teaching tactics, when programming is taught as content rather than as a tool, are likely a result of this regulatory decision. Having programming as a separate but compulsory subject for all students would have rendered these tactics unnecessary.

9.3 Facilitative strategies

Often, regulations need to be supported by secondary strategies to accomplish the desired effects. Common to this kind of strategies is their aim to create an environment in which it is easy or favorable for people to make choices that are consistent with policy goals. These operations are here grouped under Facilitative strategies (Figure 9.1).

Among the facilitative strategies that emerge from the data, only Educational strategies were actually carried through during the period covered in this thesis. The other two categories, Incentive-based strategies and Coercive strategies, are inferred from the interviews, where the teachers mention possible institutional approaches, (not necessarily those they would prefer).

Educational strategies were instantiated in two different formats: public awareness and professional training. Public awareness includes efforts devoted to informing about the reform and the rationale behind it, with the purpose of educating not only teachers, but also principals, custodians, and the general public. Debate panels and op-eds were common at the beginning of the reform (e.g. Larsson, 2017; Stolpe et al., 2019) but faded promptly in the pandemic whirlwind.

The second educational strategy was to offer workshops and courses in collaboration with universities together with online material with programming content and pedagogical perspectives directed to in-service teachers. de Certeau refers to this type of operation as the “Myth of education […] within
9.3. FACILITATIVE STRATEGIES

the frameworks and structures regulating cultural politics” (de Certeau, 1984, p. 166). Education, coming from a position of power is set in clear opposition to the workarounds that characterize tactical actions, even if in this case, the tactical actions of the teachers are also educative.

Educational strategies serve a second purpose in the professional development of teachers. By instantiating the requirements of the reform in examples and educational material, these initiatives bridge the gap between policy directives and classroom implementation favoring certain interpretations of the curricular directives. Evidence of this kind of effect can be gathered from the activities proposed by the Agency for Education, in which a dominant paradigm (imperative) and a particular programming language (Python) percolate from the educational strategies into the established practice. The favored interpretations are however not always uniform or self-consistent. Contradictory messages within educational strategies were unveiled in the analysis of the programming activities endorsed by the Agency for Education (subsection 6.1.3, p. 86). There one could find both advice against using programming where other tools were better suited, and also student exercises requiring the opposite. These disparities may be attributed to the diversity of actors engaged in deploying educational strategies at the outset, but also to the fuzziness in the regulative strategies.

The educational strategies that were deployed to facilitate the uptake of the reform are explicitly mentioned in the interviews, often accompanied by the cautionary tale endorsing the shortcomings of isolated initiatives.

Here we see a kind of speech emerging or maintaining itself, but as what “escapes” from the domination of a sociocultural economy, from the organization of reason, from the grasp of education, from the power of an elite and, finally, from the control of the enlightened consciousness (ibid., p. 158, quotations in original).

The teachers denounce the lack of time available for teacher professional development and wish that there were external encouragement for those teachers who choose to learn programming. These Incentive-based strategies, could be organized as financial compensation or other rewards to motivate teachers to improve their programming skills. The corresponding Coercive strategies, by which force or threats could be used to achieve those same goals are no-
tably absent from both the official strategies and the views of the teachers. The suggestion that programming training should be mandatory for all in-service mathematics teachers was dismissed as unrealistic in the short-term and considered a self-fulfilling outcome once programming began to be taught in Teacher Education.

9.4 Mitigative strategies

The meta-category of Mitigative strategies encompasses proactive measures designed to minimize the negative impact of the reform and to manage potential hazards and vulnerabilities. Two different approaches are included here: Averting strategies and Adaptive strategies.

Several of the decisions that preceded the reform could be characterized as averting strategies, whose purpose was to prevent problems from occurring in the first place. de Certeau reflects on how the administrative apparatus organizes order as a means of prevention. “[...] This prophylactic campaign was supposed to caulk up all the cracks through which the enemy slipped in (de Certeau, 1984, p. 196)”. To this category belong pivotal resolutions such as avoiding a required programming language, leaving the door open for personal preferences and future changes (e.g. C:4).

After launching the first curricular revision, there was a need to address newly exposed problems. We find here examples of adaptive strategies, with the objective of adjusting to the effects and problems that could not be prevented. de Certeau uses the metaphorical expression “Therapeutics of extraction [when] the disorder is caused by an excess, something extra or superfluous which has to be taken out” (ibid., p. 143) in contrast with therapeutics of ‘addition’ where a deficit becomes apparent and has to be compensated for by including something that was previously missing.

Many teachers were expecting a successive upscaling on the impact of programming, particularly with respect to the National Exams (e.g. I:1 and A:7). For them, once programming made its entrance into the unforgiving halls of assessment, there would be a motive and a goal to accelerate the process. Instead, the Agency for Education opted for ‘extraction’ and resolved to remove programming from some of the mathematics courses and downplay its importance in others, effectively attending to the actual uptake of the reform.
9.4. MITIGATIVE STRATEGIES

The practical wisdom of making decisions based on general principles conforms to the aristotelian *phronesis* doctrine that was explained in section 4.3. This concept was connected to the way policy could impact practice through authority and education. Here, the strategies upholding policy were found to belong to similar categories (regulative, facilitative, and mitigative). Policy can both limit the available options for immediate action and encourage the development of new principles that align with teachers’ understanding of their practice. This view augments the agency of teachers, beyond functioning as mere implementers of educational reforms, to include also their transformative decision process (Giroux, 2018).

In the context of policy networks, it has been posed that teachers might have little agency in determining ‘what can be said’ and ‘what can be thought’ about education policy (Ball et al., 2011, p. 611). I would instead argue that while teachers are vocal about their disagreement with policy, they are also compliant with the directives, they navigate policy as discourse and policy as text, —which are not always a coherent unit of statements—, and create a collective practice enacting shared beliefs and their individual tactics.

This chapter has discussed the strategic vision behind the introduction of programming in mathematics. Three meta-categories were devised with the intention of explaining the institutional measures towards the implementation of a revised curriculum. Combining the ideas from chapters 6 and 7, the strategies are intertwined with the resulting tactical consequences to bring together the realms of policy-making and policy implementation. The categories here address therefore no more than the data gathered for this thesis and lack any ambition of being comprehensive or exhaustive in other contexts. Nevertheless, the model offers explanatory prowess that might enhance our understanding of similar strategic endeavors.

The following chapter (10) will continue the path of policy enactment examining the two tactical types of operations disclosed in chapter 7 and addressing the next question: *How did mathematics teachers adapt their practices in response to the addition of computer programming to the curriculum?* (RQ2).
Chapter 10

The role of programming in mathematics

Nine participants offered examples and reflections to illuminate the renewed role of programming in learning mathematics in line with the inquiry that directed the investigation and gave meaning to the findings: How did mathematics teachers adapt their practices in response to the addition of computer programming to the curriculum?

10.1 The trade-offs of teaching mathematics with programming

Once the curriculum revision was rolled out, mathematics teachers in upper secondary schools were expected to include computer programming to advance their students’ problem-solving skills. During the first years after the reform, many pieces needed to fall into place. The research ground was thin regarding how and what to teach to achieve the goals of an integrated curriculum and most teachers needed to learn to code themselves before being able to integrate programming in their lessons (Misfeldt et al., 2019). By choosing mathematics teachers who were also proficient in computer programming, this investigation eluded the second obstacle and was able to focus on the first dilemma. While the question of how and what to teach has yet to be satis-
factorily answered, this thesis provides examples of teaching practices that
in different ways address the inclusion of computer programming in upper
secondary school mathematics.

The traditional policy of separation between programming and mathemat-
ics subjects was reviewed and programming was blended into a new mathe-
matics curriculum. Programming had made an entrance as a problem-solving
tool in mathematics but at the same time, it persisted as a separate elective
subject with its own curriculum and its own learning goals. By ‘having it both
ways’, keeping computer programming as a separate elective subject and at
the same time including it in compulsory mathematics courses, the message
was ambiguous and inconclusive. How much programming is necessary to
solve mathematical problems? To what extent is it a tool rather than an inde-
pendent science?

Another dimension that affects the chosen tactic is the chronic lack of time
that marks the teaching profession, from tight schedules and overstuffed courses
to scarce PPA\(^1\) and digitalization demands. Many teachers feel that program-
ming is being shoehorned into already full mathematics courses without spar-
ing any other content and that consequently, the necessary planning time has
increased. This is reflected in the unavoidable prioritizing and short-cutting
tactics that characterize both dual teaching and interspersed programming.
The two options are hardly reconcilable in a situation in which both teachers’
time and pupils’ programming skills are scarce resources. Once it was clear
that the challenges in the uptake of the reform would prevent programming
from being tested in the National Exams, prioritizing other aspects of mathe-
matics became a reasonable choice.

In summary, the introduction of programming in the mathematics cur-
riculum for upper secondary education has the capacity to change the way
teachers plan and enact their mathematics lessons in two directions. In some
cases, two parallel and yet uneven progression paths are followed, with pro-
gramming being relegated to time-filler that is resorted to once more urgent
mathematics content has been dealt with. In other cases, coding activities
are scattered throughout the course by reason of their instrumental value for
advancing mathematics but without counting on students learning program-
ming in a structured manner.

\(^1\)Teachers’ planning, preparation, and assessment time (PPA)

142
10.2 Teachers’ tactics amid institutional strategies

One of the main ideas explored in this thesis refers to the teachers’ practices interpreted as tactics that to some extent diverge from the intentions of the curricular revision. The analysis draws on the theories of practice developed by Michel de Certeau. In his metaphor of consumption, curriculum could be considered an institutional product to be ‘consumed’ by teachers. Rather than a passive use, tactics emerge when the teachers attempt to mediate the meaning and utility of the new curriculum by means of creative adjustments that include their preferences and views. de Certeau’s detailed description of tactics as different types of operations in the space of the other and the exemplifications provided later by his adepts serve as a frame of reference to understand teachers’ actions in this new reality (de Certeau, 1984; de Certeau et al., 1998).

Swedish teachers enjoy some degree of autonomy to choose teaching activities within the directives that regulate their profession and the constraints of their particular school setting such as textbooks, students’ devices, or time-schedule. For the mathematics teachers who collaborated with this research, the introduction of programming is seen as an alien element that made them surrender to tactical practices. In this, they try to create a new norm, adapting the available structures to work for their own teaching goals.

Data from the cohort of mathematics teachers proficient in computer programming that informs this study reveal two particular tactics in response to the new mathematics curriculum: dual teaching and interspersed programming. Dual teaching and Interspersed programming reflect the ontological beliefs along the science-tool dimension, or as it was previously suggested, the difference between planning mathematics activities with elements of programming and planning programming activities with elements of mathematics (p. 101). Those tactics were necessary in order to negotiate instruction in a conflicting setting that prevented the full potential of the reform from flourishing. The dichotomy appears in the ways that mathematics teachers envision their unit plans to match with the objectives in the new curriculum.

The characteristics of these tactical moves are discussed here in an attempt to pinpoint the implications that those practices could have for this and future reforms.
CHAPTER 10. THE ROLE OF PROGRAMMING

Teachers resorted to a dual teaching tactic to uphold existing course structures from previous years,—same lesson content, same textbooks—and at the same time grant their students the possibility to learn computer programming. This implied tactical compliance with the curriculum, by which students were given the possibility to solve mathematical problems with programming, even if those problems did not pertain to the syllabus of the mathematics course they were studying. In this view, teachers decide upon a content progression in each of the subjects independently, attending to the demands of the mathematics units to be covered on one side and the needs of computer programming beginners on the other. From the interviews, it was found that the teachers often recognized the limitations of the solution and justified it by pointing at students’ lack of programming skills as a consequence of the rapid implementation of the new curriculum. The approach is understood as a tactic of *bricolage* in that the teachers rearrange the available materials—timeslots in the schedule, available devices—to create new constellations that make learning programming possible. They also play with mathematics content material from various sources to construct programming activities that are familiar to the students and belong in the subject matter, reusing existing structures in what de Certeau would call *making-do* (de Certeau, 1984, p. 29). In pursuit of a programming progression that could facilitate student active learning, dual teaching sacrifices the internal coherence of the mathematics subject and perpetuates the subject separation.

Interspersed programming is a different kind of tactic, more subtle in its intentions but still powerful in its results. The teacher who chooses interspersed programming enjoys coding and is familiar with advanced features of the programming environment such as libraries, graphic interfaces, or file protocols. They produce creative solutions that fit a specific purpose in the course and prepare for reusing the new material in upcoming years. The unit plans that translate the curriculum into course actions appear impervious to the changes, but the lessons are sprinkled with programming examples. The prescriptions of the curriculum are met to the letter, in what could be seen as compliance. The teacher is aware of their exemption to actually teach programming and uses this as a loophole to avoid confronting students’ current programming knowledge; instead, they concentrate on the mathematics content that they want to address. This is an example of *la perruque*, an oppo-
sional tactic in “which the worker’s own work [is] disguised as work for the employer” (de Certeau, 1984, p. 25). Teachers designing their mathematics lessons in an interspersed programming fashion might renounce students’ autonomous coding in order to transmit the relevance of programming as a problem-solving technique and as a means of investigating the current mathematical topic. In these cases, a pre-planned sequencing of mathematics solely determines the proposed learning activities, with programming examples of different levels of sophistication sprinkled among them. Students’ engagement and participation in the programming activities might vary depending on their previous programming knowledge but the teachers’ focus would remain on making the mathematics concepts comprehensible.

Common to both dual teaching and interspersed programming were other concealed tactics of harnessing, holding back the full deployment of the reform in all its truth and breath. Teachers refrained from putting too much effort into adapting to the new directives because they expected them to be volatile, as this and other reforms have proved to be. Tactics can thus be considered beyond the idea of misalignment with curriculum and instead recognized as containing practices that could inform further strategic decisions.

This seems to suggest that teaching practices can be tactical at different levels, overt or concealed, intentional or accidental. Furthermore, the results of the second iteration display the presence of a productive middle ground in which Dual Teaching and Interspersed Programming overlap and bring forward new forms of mediating the inclusion of programming in mathematics.

In both cases and in the in-betweens, de Certeau’s theories provide teachers and education researchers with a unique point of departure for rethinking the implications of policy in everyday practice. Curriculum implementation from the point of view of teachers’ professional practice is the art of reclaiming the textual material “making it one’s own”, appropriating or reappropriating it, in a fashion that serves the interests and beliefs of the practitioner (ibid., p. 166). This appropriation is not dishonest but necessary and generative. It adds value to the syllabus by reifying the core elements of mathematics and programming into new learning activities.

For curriculum designers, the results imply new insights into what teachers see as the kernel of their subject and how this kernel evolves towards or resists against the inclusion of new technologies. As a spillover of the work with
teachers’ classroom practices, pedagogical experiences of learning mathematics with programming emerge that could be inspiring for other teachers and teacher educators. This includes new ways of understanding programming in mathematics that transcend the instrumental agenda of the curriculum. The findings show that for mathematics teachers, the place of programming is not only to be a problem-solving technique but in a wider sense, to illustrate mathematical ideas in the form of analogies and demonstrations. This view, in line with what Rowland (2013) classifies as transformation, needs to be acknowledged in future curricular reviews as it already is in the practical teaching materials (see subsection 6.1.3).

With respect to teacher training initiatives in programming and programming didactics, the two overarching tactics disclosed in this study pose a relevant issue to be addressed. A programming course for mathematics teachers at a beginner level is bound to follow a sequence that reflects primarily the cognitive complexity of programming, given that mathematics knowledge can be assumed. This is even more so if the course is open to teachers in other subjects, in which case, subject didactics become a private matter (eg. Statens Skolverk, 2017). The risk here is that, in the manner of cultural memes, the progression by which teachers learn programming is perpetuated in the way they subsequently teach programming to their own students. Furthermore, their choices will influence the way in which their students understand the subject(s) and how they eventually reproduce this knowledge. To open for wider programming didactics, teacher educators need to be aware of the dilemma and develop solutions to handle it. The educative process should provide learners with tools to recognize and value different tactical practices and their creative potential. In this endeavor, the examples provided in the present work can serve as guidance and inspiration in teacher education programmes as well as in-service teaching training. The competencies and creativity needed to engage mathematics with programming are already there to be discovered and fostered in teachers’ everyday practices. As de Certeau himself writes, “it is always good to remind ourselves that we mustn’t take people for fools” (de Certeau, 1984, p. 176).
10.3 Strategies and tactics in times of change

In the present thesis, identifying and analyzing the tactics and strategies that surface in times of institutional change provides a new perspective to understand how practitioners adopt and adapt to externally mandated developments. With de Certeau’s theory, the narrative is augmented with reference to the broader context. The tensions that govern authority and autonomy become the focal point for investigating a process of change. The intentions and objectives of the institution are then explicitly articulated in strategies to be operationalized in practice. The practitioners’ own goals and beliefs become the concrete yardstick against which they can evaluate the plausibility of the new requirements and the gap gives rise to tactics.

The theories presented by de Certeau in his seminal work *The Practice of Everyday Life* are articulated through two essential concepts: strategies and tactics. Understanding this dichotomy and related terminology establishes the base of a deeper analysis of the tensions between power and resistance, between regulations and autonomy, in short, between institutional and quotidian life. The significance of de Certeau’s understanding of everyday practices in terms of strategies and tactics is elevated to a working theory above the particularities of programming in mathematics and into the more general realm of institutional change.

Reasoning in terms of strategies and tactics serves to understand mathematics teachers’ choices in the presence of a curricular transition enforced by an external power. However, in de Certeau’s theory, power is not unidirectional but complex in its very nature, subject to being reconfigured and subverted “in the very victories it seems to have won” (de Certeau, 1984, p. 32), granting greater agency to the weaker party in the asymmetrical relations that enact authority. It is in the everyday practices that conquests and defeats take place, where democracy and emancipation are exercised against rigid control and compliance. The educator must understand and appropriate these practices and the tactics that they could entail. Once the tactics deployed by the teachers are no longer hidden transcripts in unit plans but open and analyzed, once conformity and resistance become visible, the positions may be renegotiated and new power strategies can emerge.

The strategy and tactic parameters from de Certeau’s work usefully under-
CHAPTER 10. THE ROLE OF PROGRAMMING

score the extent to which teaching practices can be explained in terms of the creative nature of teaching activities that make use of curriculum as an educational product. For the tactics of teachers to affect the strategies of institutions, we should acknowledge that what teachers produce within the constraints they operate results in new structures and offers a helpful vision of a way forward. As educational boards propose new reforms and curricular revisions, teachers’ tactical approaches should be taken into account. Thus, it is pertinent to re-evaluate and de-romanticize the tactical practices of resistance and strategic exertions of dominance to include the more mundane relations between practitioners and institutions, their discrepancies and reconciliations, in a fluctuating accommodation of interests. The contribution to the design and implementation of new learning standards will hopefully come from renewed solidarity with both teachers’ everyday struggles and policymakers’ divergent interests. Observing the dynamics of power and subversion could conceivably lead to meaningful consensus and shared goals among the educational community in future curricular reforms. When curriculum is understood as a strategic product, its demands are weighed against the potential benefits and tactics arise to tip the balance. Such a reconceptualization shifts the focus from strategies of enforcement toward strategies of cooperation, in which the intentions and mechanisms of the new directives are made comprehensible to the teachers themselves and guide their everyday practices.

The role of institutions is twofold. On one hand, institutions are characterized by stability and system reproduction maintained by means of obligations and regulations. On the other hand, institutions need to manage change and act proactively. Change in the institutional scenes is a pervasive element that both derives from and leads to development and innovation; a phenomenon that can be observed in all from system transitions, new platform implementations, automatizing routines, or communications upgrading. Common to these change processes are top-down decisions and directives originating from what de Certeau calls *a proper place*. The practices of everyday life, he adds, are the operations of those subject to act in the space of others. Everyday practices are the arena in which tactics take form and thrive. As Brewer and Werts (2017) explain, it is in times of institutional transition when tactics and strategies became most visible. Institutionally prescribed changes such as new curricula provoke a chain reaction of adjustments in strategic
10.3. TACTICS AMIDST CHANGE

policy revisions and tactical responses in search of a fruitful equilibrium.

Four years later, the pioneering practices might not have settled into a yearned-for fruitful equilibrium but a distinct pattern of practice has begun to emerge. Teaching mathematics with programming is less an improvisation act and more a calculated choice. In chapter 8 the teachers showed that certain teaching activities remained consistent while others were abandoned. The next chapter (11) discusses how incorporating programming could form the basis for an enduring mathematics teaching approach and why, rather than rebranding an existing idea, programming offers genuine opportunities to transform the mathematics classroom.
[...] a “style” that articulates itself into practices, defining a *modus loquendi* and/or a *modus agendi* [...] What is essential, then, is not a body of doctrines [...] but the foundation of a field in which specific procedures will be developed: a *space* and an *apparatus* (de Certeau, 1995, p. 14, italics and quotations in original).

Michel de Certeau has served as a provocative, if perhaps enigmatic, interlocutor in the exploration of policy strategies and tactical decisions. Turning the attention to his views on practice, the reader seems to be left to the intuitive meaning of the concept, grasped nonetheless by numerous mundane examples. A pedestrian that walks, a neighbor that cooks; practice is an expression of choice in this often unresolved imagery. We get to know that practices are, “ways of operating or doing things” but also “something constantly slipping away” (de Certeau, 1984, p. 77) which meaning is inscribed by readers/residents/pedestrians through use rather than words. de Certeau acknowledges that “attempts to capture practices will always return [to] the manner and limits of language” (ibid., p. 11) and this is also the case for the teaching practice that occupies the following discussion.

We turn therefore to the teachers’ mindful inspection to learn from past teaching experiences, reaching beyond incidental informed choices and transforming long-term teaching practice (Archer, 2012). Practice here is purposely condensed to the teachers’ perspective, and to their planning and conscious decisions with respect to their subject as a point of departure, what de Certeau
called “permanent practices of thought” (de Certeau, 1984, p. 203). It leaves out other not less important aspects of practice, such as the more immediate interactions with students and peers or the cultural models in which teaching is enacted (section 4.3).

The official strategies and tactics explored in the preceding chapters shall be understood within a fluctuating historical landscape. The disruptive events that marked the years after the first revision of the curriculum were indubitably in the background while a new practice of teaching programming in mathematics was being formed. Nevertheless, the analysis of the data collected for this study is guided by the continuity aspects revealed in the interviews. In the first five years following the introduction of programming in the mathematics curriculum, the activities in this realm have been gradually stabilizing, converging towards sustainable teaching practices. This chapter aims to discuss the enduring effects of the initial implementation efforts; the practice beyond the initial novelty.

The chapter is organized in three sections. The categories in chapter 8 lay the foundation for a discussion on how mathematics teaching can accommodate programming (section 11.1). The following sections, analyze the absence of technology and the meaning of accepting a new technology-mediated practice.

### 11.1 Forging mathematics teaching practice in the presence of programming

In the antechamber of the third curricular reform, programming might have found a sustainable place in mathematics teaching practice. The results from chapter 8 indicate that the teachers have settled significant guidelines that reconcile the external conditions of the reform with their own intentions. The motives that prompted them to include certain programming activities and reject others were often explicitly mentioned during the interviews, but can also be inferred from the comparison to initial activities.

Clarity in learning goals is one of the salient insights on which the teachers have reflected. Having experimented with programming in mathematics for some time, the teachers have become aware of which skills they want their
students to master and how to convey the necessary knowledge for that purpose. This is not to say that the teachers were previously oblivious to the learning outcomes of the chosen programming activities but rather that they had been actively learning from earlier trials to refine the activities and reduce unnecessary complexity.

An implicit consequence of this decision is gleaned from the form of the programming activities, which seem to have moved away from those common in conventional programming education. Instead of favoring independent coding exercises, mathematics teachers prefer to bring programming into the classroom by means of demonstrations and tinkering, which are cognitively easier for the students. This decision could be a reaction to the vague requirements in the curriculum and might be also due to the diminished scope of programming after the second revision. As Hanne points out, there is a narrow difference between “‘using code’ or just ‘an example of code’” (H:2). Both tinkering and demonstrations are possible schemes to fit the stipulations and both can accommodate the needs of students with limited programming knowledge.

Another slight shift in the visions that guided the teachers’ activities with programming was the emphasis on the practical relevance of programming by connecting it to real-world situations. The teachers take advantage of how programming allows for tangible applications in mathematics that complement the theoretical knowledge.

A central belief was that the purpose of programming in mathematics was to understand and explain mathematical patterns. Programming would help to reason about mathematics and to actively construct creative solutions to mathematical problems. The teachers saw many connections between the two subjects and were keen to point out the fact that the purpose of mathematics was to be applied, to solve problems. There were examples of both an instrumentalist focus on performance, under the metaphor of a toolbox, combined with a more platonic view of mathematical objects whose abstract existence provided beauty comparable to that of music or poetry. Programming in mathematics was for those teachers a gateway to provoke students into exploring and conjecturing, in a process that was creative and useful.

Despite the well-established learning potential of creative activities, the autonomy of students in the programming exercises seems to have been re-
duced. In the initial years, the teachers designed activities that could be adjusted according to students’ varying levels of programming knowledge. However, in the second iteration, there was no mention of such adaptability. The change suggests a move towards curated activities accessible to all students, possibly driven by the fact that the level of programming among new students had remained quite low and the time devoted to programming had decreased.

The teachers highlight the importance of equity and inclusiveness. Because of the constraints that surround the implementation of the reform, their intention boils down to uniformity in programming activities, designed to ensure that all students, regardless of their programming backgrounds, can participate and succeed. While open-ended programming exercises hold potential for some students, the teachers consider structured activities more suitable for reaching a broader range of students, especially when time is limited.

The focus has shifted towards simpler programming activities, leaving out complex tasks, particularly those that could involve devising algorithms. Algorithms are one of the central objectives of programming education and lie at the core of even the most elemental programming courses, even before novice programmers move over to text programming. The complexity score for block programming that was described in section 3.2 on Learning progression for programming assumes that the children solving the problems are creating algorithms (Table 3.1). When the classroom activities are confined to demonstrations and controlled tinkering examples, it can be challenging for the students to learn the rudiments of algorithmic thinking. Going back to the metaphor of language acquisition (subsection 3.1.2), it is difficult to learn to speak a language if one is only listening.

The question of assessment appears to have been settled. Programming was found to be excluded from the National Exams, thereby excluding it from other related tests as well. This established understanding regarding assessment relieved the teachers from the burden of deciding what kind of programming was to be tested and how such an assessment should be conducted.

The most notable absence was however the technology aspect. Surprisingly, the teachers expressed considerably less concern about technical issues during the second iteration. Factors such as the choice of programming language and the availability of computers were not longer addressed, possibly an indication of students’ growing comfort with the technological require-
ments but also the teachers’ deliberate focus on the intrinsic purpose of including programming in mathematics teaching. These ideas will be discussed in more detail in the following section.

11.2 Practice acceptance beyond technology

Theories on technology acceptance have guided many system design investigations during the last three decades, producing explanation models that attempt to predict the uptake of digital technologies in organizations. In the era of postdigital education, where the distinction between digital and non-digital may become inessential to the design and implementation of learning activities, we face the challenge of explaining, predicting, and promoting new school practices that go beyond technology. In this endeavor, Practice Acceptance appears as the next step to capture the nature of acceptance in relation to technologies, in which the artifacts themselves are not the object of acceptance or rejection, but rather the practices that a technology class brought about. This section raises the discussion about whether programming could be considered a technology or a practice, along other pedagogical practices of mathematics teachers.

Acceptance is generally understood in opposition to resistance or unwillingness. In literature, acceptance has been classified into five ordered categories, ranging from shallow adoption to emotional connection, willingness, and joyful use (Adell, 2007). Teo and Van Schalk differentiated acceptance from support meaning that acceptance relies on the willingness to capitulate to some external constraint (e.g., a new curriculum) while support includes appreciation and even pride and satisfaction in doing so (Teo and Van Schalk, 2009). In both cases, the concept is inherent to the individual, their personal attitudes, expectations, and experiences. Acceptance comes therefore from the teacher’s own evaluation of the technology and the idiosyncratic benefits of using it. Naturally, those advantages will only influence the level of acceptance if they are known, understood, and trusted by the teacher. For an educational technology to prevail it needs a coherent and logical foundation in a theory or theories of learning and this foundation must be properly understood and communicated (Maddux and Johnson, 2011). In our context acceptance is viewed as the degree to which a mathematics teacher chooses to
teach mathematics with programming activities.

In studies addressing Technology Acceptance Models, technology is presented in two very different costumes. It is either a specific digital system (a chat robot, a virtual doctor, a programming language ...) or it is blackboxed under the assumption that we all share a common understanding of what the abstract concept of technology entails. Technology includes all human-made artifacts with the purpose of enhancing the outcomes of an activity. However, in TAM literature, there is a clear inclination to presuppose that technology refers to—or at least includes—digital components.

Another prevailing trait found in many of the studies that use the Technology Acceptance Model is the fact that the technology is expected to enhance some activity that was already taking place before the introduction of the artifact, i.e., technology is seen as a service to fulfill a greater purpose, not an end in itself. Those studies take a stance on the basis of an alien technology being introduced in an otherwise functioning system to improve a certain task or overcome perceived difficulties. Here we find important differences in the way policy and profession understand the addition of programming to the mathematics subject. For policy, programming is seen as a technology that can improve students’ learning in general terms, by bringing education closer to the global digitalization goals of the 21st century. In this view there is no particular task which programming is set to improve, but rather education as a whole. For mathematics teachers, on the contrary, programming becomes a specific addition to their subject, with the connotation that learning in mathematics will benefit from it. This reflection perpetuates the idea that the teacher’s methodology and the curriculum to be taught precede and subdue the technology that is to be deployed. In the context of the relation between computer programming and mathematics in the curriculum, the narrative revolves around terms such as “integrating” or “incorporating”, that is, an addition to the existing curricula. However, for the teachers in this study, programming in mathematics is about a new teaching mindset, a new practice built upon the ideas behind computational thinking.

The discussion above leaves an open question regarding whether the introduction of computer programming in the mathematics curriculum could be fruitfully analyzed within a technology acceptance framework. The crucial issue is to which extent programming knowledge could be considered
11.2. PRACTICE ACCEPTANCE

a technology. Is computer programming an artifact to which one could assign certain materiality? Materiality deals with properties of the software—the technology—, that allow users to execute some action. The materiality of software is the result of previous programming activity, but the programming activity itself seems to escape the scope of materiality. Even more so, programming in mathematics is embedded in a larger learning dimension that reaches beyond any particular proprieties of the software being developed in the classroom. The technology is no longer the object that needs to be accepted, but only one of the factors allowing a whole new practice to be embraced. The ‘code processing’ artifact is not in focus, but the coded artifact is; the program and its outputs. The traditional variables of technology acceptance, namely a seamless experience, compatibility, usefulness, confirmation, satisfaction or enjoyment are not what the teachers intend in their lessons, neither for the code processing artifact nor for the program and its outputs.

The programming education that the mathematics teachers constructed aligns with this view. It is less concerned with teaching students to interact with ICT artifacts or mastering technicalities and emphasizes instead the generation and interpretation of information by means of computation. The core of Information Systems lies in the human capacity to interpret and make meaning from the output of technology. The practice that the teachers cultivate will hopefully help students see the broader significance and implications of the information generated by technology.

Moving the focal point from technology to practice allows for a different analysis that emphasizes the importance of context, learning, and change. Accepting programming as a practice that belongs in mathematics should therefore zoom out from an individual acceptance to a broader community acceptance. As a consequence, such a shift might allow for explanatory models that predict the acceptance of a new practice in a community, not merely as the sum of individual acceptances but as a social recognition of behavior.
Chapter 12

Thesis Summary

Entering the post-digital era, we anticipate that the ideas of disruptive technologies will be left behind to embrace the materiality of artifacts—their affordances and limitations—regardless of their underlying construction. In this line of development, technology is embedded into education to reach learning goals beyond the mastering of the technology in itself. Curricula around the world are acknowledging this view to broaden pupils’ skills and subject understanding. In particular, learning mathematics is moving away from learning mechanical procedures toward visualizing functions, analyzing large sets of numerical data, devising algorithms to solve problems, and emulating stochastic scenarios. Computer programming serves as a means to achieve those goals and many mathematics teachers are now expected to include coding as yet another tool in their teaching arsenal. Towards a better understanding of the opportunities and consequences of this shift, the present work contributes with a teacher-centered analysis of programming activities in mathematics.

12.1 Outcomes of the investigation

This thesis was concerned with issues of mathematics teachers’ decisions in Swedish Upper Secondary Education regarding the addition of programming to the national curriculum of their subject. In my investigations, I sought to identify three courses of action: the strategies involved in the policy pro-
cesses surrounding the curricular reform; the tactical positions that the teachers adopted in return and over time; and the emerging practices regarding programming activities in mathematics teaching.

In answering these questions, the thesis illustrates the bidirectional adjustment of practices, both being shaped by and in response to the new directives. It explains how the curricular revision and its diverse administrative tentacles frame a new teaching context to which teachers adapt and respond. To this endeavor, the thesis also provides new knowledge in adjacent areas. These contributions are here summarized and contextualized.

A map over the curricular reform

The document analysis in section 5.2 and chapter 6 serves as a comprehensive guide to navigate the landscape of the latest curricular reforms and their implementation in Swedish schools, with a particular focus on programming in mathematics in Upper Secondary Education. Through a meticulous examination of the initial version and the subsequent revisions, the research shows how the reform unfolded, pinpointing the key elements that shaped the direction of its execution. Of particular influence is the lack of a sanctioned progression regarding programming which could have established clearer prerequisites for the students entering Upper Secondary Education. Instead, the teachers were responsible for handling large disparities among their pupils’ prior programming skills. In answering this concern, both the policy documents and the enacted curriculum in the National Exams keep the exit requirements vague as well, leaving the door open for particular interpretations and ambitions.

The uncertainty with regard to the underlying technology, such as a preferred programming language or a computation paradigm, seems to have been tacitly settled among teachers and other stakeholders, such as textbook publishers. Imperative programming —where the programmer specifies a sequence of steps that the computer must follow to execute a task— remains the undefeated paradigm with Python being by far the most common programming language in Upper Secondary Education.

By delving into policy documents, this work unveils a detailed cartography of the reform, offering educators, policymakers, and researchers a clear understanding of the curricular changes, thereby contributing to a nuanced
comprehension of the dynamics involved in educational transformation.

**An analysis on how teachers understand programming in their teaching practice**

The investigation revolved around mathematics teachers facing a curricular revision in a teaching context in which they could exercise some degree of autonomy within the accountability systems in place. Their choices and creative solutions show how they navigate stipulated policies seeking a balance between the external constraints and their personal views on teaching and learning. Their independent decisions are not necessarily defiant but the result of everyday practices amid conflicting circumstances and contradictory guidelines. The thesis gathers a varied collection of programming activities suited to be included in mathematics syllabi across programs and courses (e.g. sections 7.2, 7.3 and 8.1). For teachers and teacher students, those could serve as inspiration as well as valuable resources for enhancing and reviewing their pedagogical tools.

**A new perspective on de Certeau’s theory of everyday practices applied to teaching mathematics with programming**

de Certeau’s concepts of tactics and strategies are used to reveal the tensions between policy and enactment and to answer to which extent teachers’ practices aligned with the expectations in the curriculum (chapter 7). Different tactical solutions —bricolage, la perruque and harnessing—were necessary in order to bridge the gap between the expectations of the curriculum and the actual circumstances in the classroom. Teachers that used ‘programming as a method to solve mathematical problems’, aligned their teaching in accordance with the directives, at least on the surface. The ambition of the reform regarding students’ programming skills was at best postponed for upcoming student cohorts.

**A stratified framework to analyze strategies**

Chapter 9 dissects the multifaceted realm of strategies employed in shaping the present curriculum reform either by their presence or their conspicuous
CHAPTER 12. THESIS SUMMARY

absence. The framework serves to clarify how curriculum policy can diversify its power in regulative, facilitative, and mitigative strategies to address the challenges between institutional vision and classroom practice. The interplay among regulatory decisions, educational support mechanisms, and proactive measures offers a nuanced understanding of how programming is expected to become an integral part of the mathematics curriculum. Furthermore, the structure can be used to analyze and compare different curricular reforms in terms of the strategies deployed.

Insights into the durable effects on practice concerning programming in mathematics

The addition of computer programming has brought forward new mathematics teaching that emphasizes concepts of simulation and iterative calculations. Also, the idea of devising and implementing stringent algorithms that mirror mathematical procedures is being embodied in the activities that the teachers organize to incorporate programming in their subject. The teachers were particularly concerned about scarce course time in relation to the content and the uncertainty regarding students’ prior programming knowledge. The lack of programming support in the mathematics textbooks and the notorious absence of programming requirements in the National Exams justified to some extent their choices and priorities.

The teachers in this study adopted two different tactics to cope with the constraints that surrounded the introduction of programming in the mathematics curriculum for Upper Secondary Education. Some teachers resorted to Dual teaching by providing their students with a reduced programming course in parallel with the traditional mathematics content. In these cases, the topic sequence was determined separately in each subject. Other teachers opted to implement interspersed programming, in which coding was added to their teaching palette and programming activities occurred where they were expected to advance the mathematics topic.
12.2 Restrictions and limiting factors

The premises of this work were to analyze the practice of mathematics teachers whose programming knowledge allowed them to implement the new requirements without further training. Their programming knowledge was initially self-assessed and later in the interviews largely verified. This precondition granted access to actual programming integration practices without waiting for the effects of ongoing teaching development initiatives. On the other hand, it precluded observations of the reform on a greater scale, when the vast majority of the mathematics teaching force conceivably embraces the addition of programming to their subject. This restriction could on the other hand be valuable as a benchmark against which to compare the repercussions of teacher training activities directed toward the integration of computer programming in mathematics.

It is worth noticing that at the time the empirical data was collected, the revision was at its earliest stages of implementation which means that students in the third year were presumably as new to programming as those who just entered Upper Secondary School. For this reason, the progression analysis of programming across mathematics courses concerns only the official documents (the strategies) whereas, for individual teachers (the tactics), the level of cognitive difficulty (chapter 3 is examined within each course, or within a single unit plan. In cases where a teacher maintains a group of students through consecutive courses, it would be possible to conduct a longitudinal study to monitor how (or if) more advanced programming was being introduced along with more advanced mathematics. While these data have not been collected in the present study, the findings regarding the prevailing strategies are still considered relevant to inform the observed tactics within the time constraints of the data collection. A larger cohort and a longer time span for the investigation should also include some perspective on student learning, attitudes, and preconceptions. Without data in this area, questions about acceptance and effectiveness among learners were inevitably left unanswered.

The Knowledge Quartet is a framework initially intended to be used for feedback and assessment of prospective teacher’s development of mathematics teaching. The idea in this study was to use the Knowledge Quartet for an expanded mathematics and programming subject and to use it to under-
CHAPTER 12. THESIS SUMMARY

stand in-service teachers’ practice. To underscore the integration of subjects, the focus was primarily on the connection category. This leap from the original design proved barren—or too difficult—for the purpose of a combined mathematics and programming teaching with the available data. This is not to say that the approach will not yield productive results in other circumstances, with more heterogeneous data or with a more open interpretation of what combined programming and mathematics entails. A concretization of the Knowledge Quartet for the new mathematics curriculum in Upper Secondary School could also be a useful tool in the development of both teacher education and in-service teacher training.

12.3 Further research

From the unit plans it was apparent that activities and exercises from the mathematics textbooks were dominant throughout all the courses. The importance of textbooks in mathematics teaching, particularly in Upper Secondary Education, has been widely documented (eg. Johansson, 2006; Sidenvall et al., 2015). Textbooks are to a great extent the tangible representations of the curriculum and many teachers rely on them to follow the content sequence and to ensure effective teaching. While the static media inherent to this format is gradually being substituted by more flexible digital solutions, their tacit normative value prevails (Gay et al., 2020). Later editions have been partially updated with programming elements to accommodate the new curriculum. In the present work, teachers showed the learning material they produced themselves. Once publishers make these kinds of exercises and activities more widely available, it is likely to have an effect on the way teachers relate to the convergent programming and mathematics content. Whether textbooks will opt for a dual or interspersed approach and which teaching tactics will prevail is yet to be studied.

In a recent report by an independent research and development institute for Swedish schools and pre-schools, the authors mention increased motivation as one of the most relevant achievements that the teachers observed in their students engaging in programming activities (Jahnke, 2020). Transfer effects were not directly studied but there were signs indicating student improvements outside programming. Students benefited from debugging tech-
12.3. FURTHER RESEARCH

Techniques and seemed to be more patient when solving problems in general, what has been called far transfer (e.g., Sala and Gobet, 2017; Barnett and Ceci, 2002; Mestre, 2006). In this line of thought, it would be interesting to understand whether the different tactics to which teachers resort are also able to increase motivation in older students and whether other capacities are improved along the way. From the perspective of teachers’ practice, it is necessary to understand to which extent transfer effects are expected and how that can generate different tactical approaches. In the overall picture of education, the mere existence of two well-differentiated teaching choices—dual teaching and interspersed programming—will generate differences between what pupils learn and how they perceive the subject. While is it probably neither achievable nor desirable to decide upon a unique teaching method, it might be necessary to evaluate how the tactical approaches could improve student’s prospects for employability or satisfactory completion of higher education.

The curriculum that was effective at the time of writing has been further revised with a new version taking effect at the start of the school year in the fall semester of 2021 and yet another one scheduled for 2024. Then, the current division of Upper Secondary mathematics in consecutive but separate courses will be substituted by a ‘subject study’ (Regeringskansliet, 2020). The new mathematics subject will be largely equivalent to the current courses in content, but the internal progression will be decided locally. What this sizable change would imply for the joined programming and mathematics teaching is not clear. Presumably, an increased autonomy to decide upon a sequence of topics during a longer period of time could lead to a new role for programming in mathematics. Whether the intentions of the curriculum become enhanced or watered down, the impact that these later developments have had in Upper Secondary Education is likely to percolate to educational research and should be taken into account in future investigations.
Every end is the beginning of something else.

References


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


References


REFERENCES


Lärarnas Riksförbund (2020). “Programmering—en skolreform utan program. En lägesbeskrivning efter två år med obligatorisk undervisning i programmering.” In:


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


— (2023). Regeringens proposition 2022/23:94 Några ändringar som rör gymnasieskolans nationella program och ämnen m.m.


REFERENCES


www.skolverket.se/download/18.4fc05a3f164131a74181063/1535372298267/Mathematics-swedish-school.pdf.
— (2016). Läroplan för grundskolan, förskoleklassen och fritidshemmet (reviderad 2018) [Elektronisk resurs].
https://www.skolverket.se/publikationer?id=3975.
https://www.skolverket.se/publikationer?id=3975.
https://www.skolverket.se/publikationer?id=2975.
REFERENCES


REFERENCES


REFERENCES


REFERENCES


# List of Tables

3.1 Programming Proficiency Levels ........................................... 30

4.1 The Knowledge Quartet .......................................................... 44

5.1 Summary of data sources for the strategic analysis .................... 54

6.1 Timeline for the regulatory decisions concerning programming in ISCED 3. ................................................................. 83

7.1 Excerpts of interviews revealing dual teaching ............................ 105
7.2 Excerpts of interviews revealing interspersed programming ............ 108
7.3 Dual teaching and Interspersed programming ............................. 112

8.1 Evolution of tactical approaches to teaching mathematics with programming .......................................................... 129

A.1 Teachers interviewed for the study ............................................ 3

B.1 Summary of proposed programming activities in mathematics in upper secondary school ................................................. 6

C.1 Invitation to tender and related proceedings from the National Agency for Education ...................................................... 9

D.1 18 National programmes for upper secondary school (12 vocational programmes and 6 programmes preparing for higher education). ....... 13
List of Figures

3.1 Transfer model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25
3.2 Debugging flowchart . . . . . . . . . . . . . . . . . . . . . . . . 27

5.1 Research journey . . . . . . . . . . . . . . . . . . . . . . . . . . . 52
5.2 Unit Plan . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60

6.1 Three levels in the inscriptionsal space conforming the strategies . 79
6.2 Bisector of a line segment. . . . . . . . . . . . . . . . . . . . . . . 83
6.3 Systems of equations with two unknowns . . . . . . . . . . . . . . 85
6.4 Syllabus for programming courses . . . . . . . . . . . . . . . . . . 86
6.5 Formula sheet for trigonometric expressions . . . . . . . . . . . . 88
6.6 Incremental approximation of a logarithm . . . . . . . . . . . . . . 95

7.1 Learning objectives for a lesson . . . . . . . . . . . . . . . . . . . . . 106
7.2 Monte Carlo simulation . . . . . . . . . . . . . . . . . . . . . . . . 110

8.1 Short and long conditional . . . . . . . . . . . . . . . . . . . . . . . . 115
8.2 Evin’s Python activity . . . . . . . . . . . . . . . . . . . . . . . . . . 119

9.1 Scheme of types of operations in strategic modalities of action.
The lighter cursive font is used to indicate those strategies that were mentioned in the interviews but have not (yet) been realized.135

D.1 Map over the Swedish education system from the Agency for Education. 12
D.2 Credit distribution in upper secondary education programmes. . . . . 13
D.3 Mathematics courses in Swedish upper secondary school (2020) . . . . 14
List of Quotations

Ariel:1 ......................................................... 63, 105
Ariel:2 ......................................................... 92, 105
Ariel:3 .......................................................... 93
Ariel:4 .......................................................... 105
Ariel:5 .......................................................... 124
Ariel:6 .......................................................... 128
Ariel:7 .......................................................... 126
Billie:1 .......................................................... 101
Billie:2 .......................................................... 105
Billie:3 .......................................................... 124
Chris:1 .......................................................... 103
Chris:2 .......................................................... 105
Chris:3 .......................................................... 114
Chris:4 .......................................................... 126
Dominique:1 .................................................... 93
Dominique:2 .................................................... 108
Evin:1 ............................................................ 94
List of Quotations

Evin:2 ................................................................. 108
Evin:3 ................................................................. 117
Evin:4 ................................................................. 125
Evin:5 ................................................................. 129
Francis:1 .............................................................. 69
Francis:2 .............................................................. 92
Francis:3 .............................................................. 108
Gaby:1 ................................................................. 94
Gaby:2 ................................................................. 97
Gaby:3 ................................................................. 108
Gaby:4 ................................................................. 109
Gaby:5 ................................................................. 123
Hanne:1 ................................................................. 94
Hanne:2 ................................................................. 97
Hanne:3 ................................................................. 97
Hanne:4 ................................................................. 109
Hanne:5 ................................................................. 120
Hanne:6 ................................................................. 122
Hanne:7 ................................................................. 127
Hanne:8 ................................................................. 127
Inge:1 ................................................................. 96
Inge:2 ................................................................. 109
Inge:3 ................................................................. 110
List of Quotations

<table>
<thead>
<tr>
<th>Quotation</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inge:4</td>
<td>116</td>
</tr>
<tr>
<td>Inge:5</td>
<td>128</td>
</tr>
<tr>
<td>Test Developer PRIMgruppen:1</td>
<td>90</td>
</tr>
<tr>
<td>Test Developer UMEÅ:1</td>
<td>89</td>
</tr>
</tbody>
</table>
Acronyms

CJEU  Court of Justice of the European Union. 89

CR  Critical Realism. 51

DT  Dual Teaching. 101

GDPR  General Data Protection Regulation. 58

GRADE  Swedish National Graduate School for Digital Technologies in Education. viii

IP  Interspersed Programming. 106

IS  Information Systems. ix, 5, 50

MWE  minimal working example. 114, 115

OECD  Organisation for Economic Co-operation and Development. 198

PCK  Pedagogical Content Knowledge. 43

PPA  planning, preparation, and assessment time. 142

TIMSS  Trends in International Mathematics and Science Study. 12, 17

TPD  Teacher Professional Development. viii, ix, 13, 83

WIL  Work Integrated Learning. viii, x, 5, 51
Glossary

course syllabus  In the Swedish Education System, the course syllabus estab-
lishes the goals covered by the current course, its scope of credits and
the core content that must be covered in terms of methods, concepts,
theories, etc. The core content does not state how much time should be
spent on the different parts. The course syllabus also includes specific
knowledge requirements for three of the passing grades – E, C and A
but does not mention assessment methods or reading materials. 79, 81,
82, 88, 90, 96

ISCED 1  Primary education provides learning and educational activities typ-
ically designed to provide students with fundamental skills in reading,
writing and mathematics (i.e. literacy and numeracy) and establish a
solid foundation for learning and understanding core areas of knowl-
edge and personal development, preparing for lower secondary educa-
tion. It focuses on learning at a basic level of complexity with little, if
any, specialization. Age is typically the only entry requirement at this
level. The customary or legal age of entry is usually not below 5 years
old nor above 7 years old. It may be referred to in many ways, for ex-
ample: primary education, elementary education or basic education. In
Sweden level 1 comprises school years 1-6. Commission et al., 2019. 55,
80, 90, 135

ISCED 2  Lower secondary education, typically designed to build on the learn-
ing outcomes from ISCED level 1. Programmes classified at ISCED level
2 may be referred to in many ways, for example secondary school, ju-
nior secondary school, middle school, or junior high school. If a pro-
gramme spans ISCED levels 1 and 2, the terms elementary education or
basic school (stage two/upper grades) are often used. For international comparability purposes, the term ‘lower secondary education’ is used to label ISCED level 2. Students enter ISCED level 2 typically between ages 10 and 13.

ISCED 3 Upper secondary education. Second/final stage of secondary education preparing for tertiary education and/or providing skills relevant to employment. Usually with an increased range of subject options and streams. U.S. equivalent: 10th-12th grades or first 3 years of vocational education. Swedish equivalent: gymnasieskolan.

There are substantial differences in the typical duration of ISCED 3 programmes both across and between countries, typically ranging from two to five years of schooling. ISCED 3 may either be “terminal” (i.e., preparing the students for entry directly into working life) and/or “preparatory” (i.e., preparing students for tertiary education).

National Exams or National tests are envisioned to support teachers’ assessment and grading. The aim of the national tests is primarily to support an equivalent and fair assessment and, to provide a basis for an analysis of the extent to which knowledge requirements are fulfilled at the school level, at the level of the organizer, and at the national level.

subject syllabus The subject syllabus describes the aim and goals of the subject in terms of “the ability to”, “knowledge about”, “understanding of” and “skills in”. It also establishes which courses included in the subject and how they are related to each other.

unit plan The groupings of sequential lessons (by theme, topic, step in a process, skill, essential question, etc.) that are components of a course. Unit plans comprise several lessons and are made to serve for a long period of time during which the topics in the unit are studied.
List of Appendices

Participants 3
Programming activities 5
Official records from the National Agency for Education 9
Swedish education system 11
Legislation and regulations 15
Appendices
Appendix A

Participants

Participants’ Demographic Matrix

Table A.1: Scheme of teachers participating in the study. For clarity and anonymity, names were assigned alphabetically according to the order of appearance in the thesis without regard to gender.

<table>
<thead>
<tr>
<th>Name</th>
<th>Region</th>
<th>School type</th>
<th>2\textsuperscript{nd} subject\textsuperscript{1}</th>
<th>math courses\textsuperscript{2}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ariel</td>
<td>Stockholm region</td>
<td>Public</td>
<td>physics</td>
<td>3c, 5</td>
</tr>
<tr>
<td>Billie</td>
<td>North Sweden</td>
<td>Public</td>
<td>physics</td>
<td>1c, 2c</td>
</tr>
<tr>
<td>Chris</td>
<td>Stockholm region</td>
<td>Public</td>
<td>Programming</td>
<td>1c, 2c, 3c</td>
</tr>
<tr>
<td>Dominique</td>
<td>Western Sweden</td>
<td>Charter</td>
<td>Chemistry</td>
<td>1c, 4</td>
</tr>
<tr>
<td>Evin</td>
<td>Mid Sweden</td>
<td>Public</td>
<td>Physics</td>
<td>2c</td>
</tr>
<tr>
<td>Francis</td>
<td>Mid Sweden</td>
<td>Public</td>
<td>Economics</td>
<td>3b</td>
</tr>
<tr>
<td>Gaby</td>
<td>Stockholm region</td>
<td>Charter</td>
<td>–</td>
<td>2c, 4</td>
</tr>
<tr>
<td>Hanne</td>
<td>Eastern Sweden</td>
<td>Public</td>
<td>Programming</td>
<td>1c, 3c</td>
</tr>
<tr>
<td>Inge</td>
<td>Southern Sweden</td>
<td>Public</td>
<td>Programming</td>
<td>4, spec</td>
</tr>
</tbody>
</table>

Summary of demographics

- Years of teaching experience: low 3 years; high 21 years
- Degree: All teachers have completed a Master Programme in Education (5 years). 1 teacher has a PhD degree.
APPENDIX A. PARTICIPANTS

- Certifications: all the teachers in the study have a national certification diploma issued by the Swedish Agency for Education and are licensed to teach mathematics in upper secondary education; 2 teachers have been appointed as distinguished teacher with leadership roles\(^3\);

- Prior training in computer programming: 8 teachers have training including university credits, 1 teacher has no academic training in computer programming, 1 teacher has professional experience in computer programming.

\(^1\)Teachers in secondary school often teach courses within two subjects, 2\(^{nd}\) in this case, refers to the other subject they teach, and does not necessarily imply priority

\(^2\)The first mathematics courses are divided into three versions, one for vocational programs (a), one for social science programs (b) and one for nature science and technology programs (c), with some differences in the content of the courses. Courses 4 and 5 exist only in one version. Mathematics specialization (spec) is a parallel course that can be pursued after finishing course 4. Computer programming is a part of the syllabus of all courses except 1a, 1b, 2a and 2b. The codes refer to the courses that the teacher was imparting that semester

\(^3\)In Swedish, Förstelärare
Appendix B

Programming activities
## APPENDIX B. PROGRAMMING ACTIVITIES

**Table B.1:** Summary of proposed programming activities in mathematics in upper secondary school

<table>
<thead>
<tr>
<th>Activity</th>
<th>Programming language</th>
<th>Course</th>
<th>Complexity</th>
<th>Complexity level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draw polygon</td>
<td>Python</td>
<td>Ma1c</td>
<td>Sequenced instructions</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Scratch</td>
<td></td>
<td>External libraries</td>
<td></td>
</tr>
<tr>
<td>2nd grade polynomials</td>
<td>Python</td>
<td>Ma2c</td>
<td>Nested constructs</td>
<td>2</td>
</tr>
<tr>
<td>System of equations</td>
<td>Python</td>
<td>Ma2c</td>
<td>Aritmetic</td>
<td>2</td>
</tr>
<tr>
<td>Distance and probability</td>
<td>Python</td>
<td>Ma2c</td>
<td>Nested constructs</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>External libraries</td>
<td></td>
</tr>
<tr>
<td>Risk</td>
<td>Python</td>
<td>Ma1c</td>
<td>Defining and calling functions</td>
<td>2</td>
</tr>
<tr>
<td>Risk 2</td>
<td>Python</td>
<td>Ma1c</td>
<td>Defining and calling functions</td>
<td>2</td>
</tr>
<tr>
<td>Euclidean algorithm (GCD)</td>
<td>Python</td>
<td>Ma1c</td>
<td>Recursion</td>
<td>3</td>
</tr>
<tr>
<td>Geometry as movement</td>
<td>Python</td>
<td>Ma1c</td>
<td>External libraries</td>
<td>2</td>
</tr>
<tr>
<td>Length of a curve</td>
<td>Python</td>
<td>Ma2c</td>
<td>Defining and calling functions</td>
<td>2</td>
</tr>
<tr>
<td>Buffon’s needle problem</td>
<td>Scratch</td>
<td>Ma1c</td>
<td>Loops</td>
<td>1</td>
</tr>
<tr>
<td>Divisibility</td>
<td>Python</td>
<td>Ma1c</td>
<td>Conditional logic</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Easy loops</td>
<td></td>
</tr>
<tr>
<td>Is N prime?</td>
<td>Python</td>
<td>Ma1c</td>
<td>Loops</td>
<td>2</td>
</tr>
<tr>
<td>Primes up to N</td>
<td>Python</td>
<td>Ma1c</td>
<td>Nested Loops</td>
<td>2</td>
</tr>
<tr>
<td>Rotate squares</td>
<td>Wolfram</td>
<td></td>
<td>Functional programming</td>
<td>3+</td>
</tr>
<tr>
<td>Collatz conjecture</td>
<td>Python</td>
<td>Ma5</td>
<td>Nested loops</td>
<td>2</td>
</tr>
<tr>
<td>Varying value development</td>
<td>Python</td>
<td>Ma1c</td>
<td>Nested loops</td>
<td>2</td>
</tr>
<tr>
<td>Reliability of test results</td>
<td>Python</td>
<td>Ma1c</td>
<td>Nested constructs</td>
<td>2</td>
</tr>
<tr>
<td>Approximation of e</td>
<td>Python</td>
<td>Ma3c</td>
<td>Nested Constructs</td>
<td>2</td>
</tr>
<tr>
<td>Fractals</td>
<td>Python</td>
<td>Ma4</td>
<td>Complex numbers in programming</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>External libraries</td>
<td></td>
</tr>
</tbody>
</table>

Continued on next page
<table>
<thead>
<tr>
<th>Activity</th>
<th>Programming language</th>
<th>Course</th>
<th>Complexity</th>
<th>level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yatzy</td>
<td>Python</td>
<td>Ma1c</td>
<td>Nested loops</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ma5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poker</td>
<td>Python</td>
<td>Ma1c</td>
<td>Nested loops</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ma5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arithmetic Series</td>
<td>Python</td>
<td>Ma1c</td>
<td>Advanced data structures</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ma5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Euler’s step method</td>
<td>Python</td>
<td>Ma5</td>
<td>Loops</td>
<td>2</td>
</tr>
<tr>
<td>Infinities and limits</td>
<td>Python</td>
<td>Ma3c</td>
<td>Defining and calling functions</td>
<td>2</td>
</tr>
<tr>
<td>Calculating with interest rates</td>
<td>Python</td>
<td>Ma1c</td>
<td>Loops</td>
<td>2</td>
</tr>
<tr>
<td>Probability and coincidence</td>
<td>Python</td>
<td>Ma1c</td>
<td>Handling Text Files</td>
<td>3</td>
</tr>
<tr>
<td>Data structures and population</td>
<td>Python</td>
<td>Ma2c</td>
<td>Handling Text Files</td>
<td>3</td>
</tr>
</tbody>
</table>
## Appendix C

### Official records

Table C.1: Invitation to tender and related proceedings from the National Agency for Education

<table>
<thead>
<tr>
<th>Event NO</th>
<th>Proceeding</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017:83</td>
<td>Digitalisering programmering samråd</td>
<td>2017-01-19</td>
</tr>
<tr>
<td>2017:626</td>
<td>Avrop gällande Nationella skolutvecklingsprogram, NSP. Digitalisering, programmering</td>
<td>2017-04-10</td>
</tr>
<tr>
<td>2017:627</td>
<td>Nationella skolutvecklingsprogram, NSP. Digitalisering, programmering, filmtillstånd</td>
<td>2017-04-10</td>
</tr>
<tr>
<td>2017:663</td>
<td>Avtal F-skatt, NSP (Nationella skolutvecklingsprogram) Digitalisering, programmering</td>
<td>2017-04-21</td>
</tr>
<tr>
<td>2017:962</td>
<td>Nationella skolutvecklingsprogram NSP, Digitalisering, Programmering, Kursmaterial</td>
<td>2017-06-16</td>
</tr>
<tr>
<td>2017:985</td>
<td>Överenskommelse med lärosäten om att anordna programmeringskurser för lärare. Fas 2 start november 2017</td>
<td>2017-06-20</td>
</tr>
<tr>
<td>2017:1486</td>
<td>Genomförandeplan webbkurs Om programmering</td>
<td>2017-10-05</td>
</tr>
<tr>
<td>2017:1538</td>
<td>Expertstöd vid framtagning av ramkursplan för uppdragsutbildning i programmering.</td>
<td>2017-10-17</td>
</tr>
<tr>
<td>Date</td>
<td>Event Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>2017:1887</td>
<td>Fas 3 Uppdragsutbildning programmeringskurser 7,5 hp</td>
<td></td>
</tr>
<tr>
<td>2017:2032</td>
<td>Insats Om programmering - en webbaserad kurs</td>
<td></td>
</tr>
<tr>
<td>2018:391</td>
<td>Inbjudan till samråd gällande programmering inom ramen för insatsen programmering</td>
<td></td>
</tr>
<tr>
<td>2018:762</td>
<td>Överenskommelse om uppdrag att ta fram material till en webbkurs i programmering för lärare</td>
<td></td>
</tr>
<tr>
<td>2018:1944</td>
<td>Fas 4A Grundläggande programmering med ämnesdidaktisk inriktning</td>
<td></td>
</tr>
<tr>
<td>2018:1945</td>
<td>Programmering med ämnesdidaktisk fördjupning fas 4B</td>
<td></td>
</tr>
<tr>
<td>2018:2006</td>
<td>Ramkursplan för programmeringskurs fas 4B</td>
<td></td>
</tr>
<tr>
<td>2019:547</td>
<td>Överenskommelse om uppdragsutbildning grundläggande programmering fas 5A</td>
<td></td>
</tr>
<tr>
<td>2019:548</td>
<td>Fas 5B ma Överenskommelse, uppdragsutbildning grundläggande programmering med matematikdidaktiskt fokus, fortsättningskurs, fas 5B</td>
<td></td>
</tr>
<tr>
<td>2019:647</td>
<td>Fas 5B tk Överenskommelse, uppdragsutbildning grundläggande programmering med teknikdidaktiskt fokus, fortsättningskurs, fas 5B</td>
<td></td>
</tr>
<tr>
<td>2019:1475</td>
<td>Expertstöd vid framtagning av ramkursplaner inför fas 6, programmeringskurser</td>
<td></td>
</tr>
<tr>
<td>2019:1727</td>
<td>Fas 6B Överenskommelser med lärosäten om uppdragsutbildningen Introduktion till programmering i visuell miljö, 5 hp fas 6B</td>
<td></td>
</tr>
<tr>
<td>2019:1728</td>
<td>Fas 6 A Överenskommelser med lärosäten om uppdragsutbildningen Introduktion till programmering i textbaserad miljö, 5 hp, fas 6A</td>
<td></td>
</tr>
<tr>
<td>2019:1886</td>
<td>Beslut och genomförandeplan för uppdragsutbildningen - Programmeringskurser för lärare, 2020</td>
<td></td>
</tr>
</tbody>
</table>

Retrieved from [https://www.skolverket.se/om-oss/kontakta-skolverket/skolverkets-diarium](https://www.skolverket.se/om-oss/kontakta-skolverket/skolverkets-diarium)
Appendix D

The Swedish education system

This appendix presents an overview of the Swedish school system in order to understand the context of this thesis. Data has been retrieved from the Swedish Agency for Education and their department for statistics (Skolverket, 2012; Skolverket, 2020b).

Particular emphasis has been placed on the intricacies of mathematics and programming in upper secondary school. Upper secondary school is the first non-compulsory education level (see Figure D.1). Almost all compulsory school pupils continue straight on to upper secondary school. Every municipality in Sweden is required by law to offer all students who have completed compulsory school an upper secondary education. There are eighteen national programs, all of which last for three years (ages 16-18). The 18 upper secondary programmes are divided into programmes preparing for further studies and vocationally-oriented programmes (Table D.1). Less than 30% of the students choose vocational programmes. All programmes provide comprehensive general education and make students eligible for studies at the university or post-secondary level.

Figure D.2 shows an overview of Swedish upper secondary school. From August 2017 programming must be available as an elective or specialization
Figure D.1: Map over the Swedish education system from the Agency for Education.

course in all National Programmes \(^1\).

Figure D.3 shows how mathematics courses are divided into three parallel lines with content tailored for the different needs of the programmes. National exams are required for concluding courses (shade). The arrows indicate possible course sequences. Starting in June 2018, programming has been a part of the curriculum in seven mathematics courses (shaded area). Around 26% of the students (81000) will take one or more courses in which knowledge of computer programming is expected (Skolverket, 2020b).

\(^1\)Föreskrifter om ändring i Skolverkets föreskrifter (SKOLFS 2017:56 - 73)
Figure D.2: Credit distribution in upper secondary education programmes.

Table D.1: 18 National programmes for upper secondary school (12 vocational programmes and 6 programmes preparing for higher education).

<table>
<thead>
<tr>
<th>Programme</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Child and Recreation Programme</td>
<td>BF</td>
<td>Building and Construction Programme (BA)</td>
</tr>
<tr>
<td>Electricity and Energy Programme</td>
<td>EE</td>
<td>Vehicle and Transport Programme (FT)</td>
</tr>
<tr>
<td>Business and Administration Programme</td>
<td>HA</td>
<td>Handicraft Programme (HV)</td>
</tr>
<tr>
<td>Hotel and Tourism Programme</td>
<td>HT</td>
<td>Industrial Technology Programme (IN)</td>
</tr>
<tr>
<td>Natural Resource Use Programme</td>
<td>NB</td>
<td>Restaurant and Food Programme (RL)</td>
</tr>
<tr>
<td>HVAC and Property Maintenance Programme</td>
<td>VF</td>
<td>Health and Social Care Programme (VO)</td>
</tr>
<tr>
<td>Business and Economics Programme</td>
<td>EK</td>
<td>Arts Programme (ES)</td>
</tr>
<tr>
<td>Humanities Programme</td>
<td>HU</td>
<td>Natural Science Programme (NA)</td>
</tr>
<tr>
<td>Social Science Programme</td>
<td>SA</td>
<td>Technology Programme (TE)</td>
</tr>
</tbody>
</table>
APPENDIX D. THE SWEDISH EDUCATION SYSTEM

Figure D.3: Mathematics courses in Swedish upper secondary school (2020)
Appendix E

Legislation and regulations

The documents steering the upper secondary school are intended to create a meaningful whole. They each fulfill an important function but also express together a common view of schooling. The Education Act takes priority over the other documents. The upper secondary school ordinance, the curriculum and diploma goals are ordinances which in different ways make the provisions of the Education Act more specific. The subject syllabuses are regulations steering teaching in a given subject.

Education Act

The Education Act contains the general provisions for all school forms and the basic provisions for the different school forms. What specifically relates to the upper secondary school is set out in Chapters 15–17. The Swedish Riksdag (Parliament) decides on the Education Act.

swedish: Skollag

Upper Secondary School Ordinance

The upper secondary school ordinance contains regulations on the upper secondary school and makes the provisions of the Education Act more specific. The government decides on the upper secondary school ordinance.

swedish: Gymnasieförordning
APPENDIX E. LEGISLATION AND REGULATIONS

Curriculum

The curriculum for the non-compulsory school forms describes the fundamental values, tasks, as well as goals and guidelines of the school. The government decides on the curriculum.

swedish: Läroplan

Diploma Goals

Each programme has its diploma goals. The diploma goals provide the foundation for planning education and teaching from the student’s very first day in the programme. These should steer the education and the organization of upper secondary work and its contents. The diploma goals set out the goals of the programme, the orientations in the programme, as well as the goals of the diploma project.

swedish: Examensmål

Subject Syllabus

Each subject has a syllabus that describes the courses included in the subject. The government decides on the subject syllabuses for the foundation subjects for the upper secondary school, such as mathematics, on the basis of proposals from the National Agency for Education. The basic principle in the subject syllabuses for the upper secondary school is that the core content primarily expresses the progression between courses.

swedish: Ämnesplan
Other Publications by the author

(Accepted) A. Fuentes Martinez et al. (2023b). “Assessment strategies for programming integrated in upper secondary education subjects.” In: Norsk IKT-konferanse for forskning og utdanning. ISSN 1892-0721


Tidigare avhandlingar — Arbetsintegrerat lärande


ANNIKA ANDERSSON In case of emergency: Collaboration exercises at the boundaries between emergency service organizations, 2016:8.

MARIE WESTERLIND Knowing at work: A study of professional knowledge in integration work directed to newly arrived immigrants, 2016:9.


LARS-OLOF JOHANSSON Engaged in digital service innovation, 2018:15.


CAMILLA SEITL Informellt lärande i en formell organisation: Om meningsskapande, kunskapsområden och kompetens i arbete med kommunala mål, 2018:19.
KARIN HÖGBERG Persistent Digital Service Encounters: Challenges of organizational use of social media in a hotel chain, 2018:22.


JENNIE RYDING Mellan evidens och reflektion: om professionellt lärande i socialtjänstens familjebehandlande arbete, 2020:36.

VILLE BJÖRK Learning 'theory' at university and 'practice' in the workplace: A problematisation of the theory-practice terminology that the dualistic design of Work-integrated Learning institutionalizes, 2020:38.


HELENA VALLO HULT Digital Work: Coping with Contradictions in Changing Healthcare, 2021:44.


ANDERS SANDBLAD Yrkeskunnande Maskiner Människor, 2021:47.

SANNA HEDREN Fritidshemmets (mellan) mål och mening: Positioneringar i kollegialt samtalande mellan fritidshemspersonal, 2022:50.

MARTIN SCHULER Säkerhetsklimat i en militär organisation, 2022:51.

SARA EKSTRÖM Teaching with social robots, 2023:55.

DANIEL OLOF WIEDEL Att skapa mål och mening i vakuumland: Om socialt arbete i en kommunal arbetsmarknadsinsats för unga, 2023:57.

ANNA ROUMBANIS VIBERG Lärarutbildarens professionella agency i en digital tid, 2023:58.

LINNEA CARLSSON Social Aspects of Strategizing Industrial Digitalization, 2023:60.
Practice beyond technology when programming and mathematics teaching converge

This thesis delves into the intricacies of teaching an integrated mathematics and programming curriculum amidst educational reforms. The reader will become acquainted with de Certeau’s theoretical foundation regarding the practice of everyday life and the power struggles at the micro-level. The tensions that emerge between policy documents and teachers’ didactical choices are generously illustrated with concrete examples of integration practices. These empirical data, together with de Certeau’s concepts of tactics and strategies, produce a new body of knowledge relevant to the field of Informatics as well as Education and Work Integrated Learning.

The thesis analyzes the types of strategies upholding the implementation of a new curriculum, offering a framework for understanding policy in terms of regulative, facilitative, and mitigative strategies. It problematizes how teachers are able to operate with some autonomy within the constraints of the new curriculum by accommodating both their teacher identity and the policy directives in creative ways. Two distinct ideas dominate teachers’ tactics when planning their lessons to include both mathematics and programming: Dual teaching and Interspersed programming. Dual Teaching builds on a pedagogical view of programming as a separate discipline with its own learning progression in which mathematics can provide useful examples. Interspersed programming, on the other hand, accounts for a more instrumental perspective on programming, a side-skill that can benefit mathematics teaching and learning. This dichotomy aligns with two ways of negotiating the addition of computer programming to the traditional mathematics curriculum by either teaching programming with elements of mathematics or teaching mathematics with elements of programming. The results unveil the difficulties of devising a common progression path for the integrated curriculum.

With a keen eye on the complexities of integration, this research offers valuable insights for educators and curriculum designers, shedding light on the subtle evolution of teaching practices and the acceptance of innovative methods in mathematics education.

Ana Fuentes Martinez

works at the School of Business, Economics and IT, division of Informatics, at University West. Her research interests revolve around the field of computer programming in education from the perspective of teachers’ training and school practices, in the environment of Work Integrated Learning.

https://orcid.org/0000-0002-1748-8837

ISBN 978-91-89325-64-7 (printed)
ISBN 978-91-89325-63-0 (digital)