Design and Technology Education: An International Journal

30.3

Assessing Students' Computer Programming Skills: How Technology Teachers in Sweden Evaluate Learning in Grades 4–6

Eva-Lena Bjursten, Mälardalen University, Sweden
Lena Gumaelius, KTH Royal Institute of Technology, Sweden
Eva Hartell, KTH Royal Institute of Technology, Sweden

Abstract

This study aims to deepen the understanding of how computer programming is taught and assessed in Swedish schools by focusing on teachers' perspectives. It explores how technology teachers (teaching years 4–6, students aged 10-12) perceive their roles and responsibilities in teaching computer programming, primarily within the technology subject, and examines what computer programming content is taught and assessed. The research is based on a survey and interviews with seven experienced teachers who taught computer programming before it became mandatory. The findings reveal similar views among the teachers but also significant variation in assessment practices, categorized into four distinct personas, ranging from a strong disciplinary content and product focus to a weaker disciplinary content and process orientation. The discussion reflects upon how these variations may be influenced by teachers' backgrounds, computer programming knowledge, and unclear policy documents. The conclusions suggest that, due to this variety, Swedish students may not be equally equipped with the digital skills needed for participating in a digitalized society. To enhance equity, we argue that teachers need better preparation to effectively integrate computer programming skills across subjects. Additionally, we recommend clearer national guidelines on how to teach computer programming and how to assess this subject content in compulsory education.

Keywords

Computer programming, assessment, technology education, teacher perspective, pedagogical content knowledge

Introduction

In an increasingly digitalized society, understanding new technology is crucial. It is essential for everyone to grasp how data usage influences their daily lives and decision-making, enabling them to make informed choices as citizens. Moreover, digitalization represents the most significant technological shift affecting current competency needs (Teknikföretagen, 2020). This underscores the importance of young individuals acquiring foundational skills that will enable them to pursue future careers aligned with digitalization.

Following EU and OECD recommendations (European Union, 2006), computer programming (hereafter referred to as programming) was introduced into the Swedish national curriculum in 2018, primarily in technology and mathematics (Swedish National Agency for Education, 2017). The aim was to foster "a general understanding of programming and its implications" (Swedish National Agency for Education, 2022a, p. 8). As a result, all K–9 technology and mathematics teachers are now required to teach this content.

Programming can be described as creating instructions for a computer to solve a problem. However, in the Swedish compulsory school context, the concept also includes creativity, control and regulation, simulation and democratic dimensions (Swedish National Agency for Education, 2022b). In this article, programming is therefore understood as a combination of these perspectives.

How teachers approach this task is critical for achieving curricular objectives and equipping students for a digitalized society. Existing research underscores the recognition of programming's integration into the curriculum across various studies, although further research in this domain is warranted (Larsson, 2023; Mannila et al., 2020; Nordén et al., 2017; Nouri et al., 2020; Stigberg & Stigberg, 2019; Vinnervik, 2021).

There are strong indications that our school students' right to equal education in programming is not being fulfilled. A follow-up report on the digitalization strategy from 2022 (Swedish National Agency for Education, 2022e) revealed that over 70% of primary school teachers (years 4–6, students aged 10-12) felt uncertain about how to teach programming, highlighting significant gaps in teachers' preparedness and confidence in delivering programming education. This suggests a pressing need for enhanced teacher training and resources to ensure that all students receive high-quality education in programming.

Through this study, our objective is to deepen the understanding of how programming is taught in schools and thereby contribute to understanding what is needed for programming education to become an integrated part of school education. We have chosen to examine our research question from a teacher's perspective. Specifically, we aim to understand how teachers perceive their responsibilities and utilize their roles when teaching programming, primarily within the technology subject. This understanding can be framed as the teachers' experienced and enacted pedagogical content knowledge (ePCK), according to the PCK framework, which is frequently employed to describe teachers' capabilities and competencies in delivering effective instruction (Carlson et al., 2019). However, PCK alone does not explain how teachers' practices are shaped by curriculum structures, subject boundaries or institutional control. Therefore, in this study, we also draw on Bernstein's (2000) concepts of classification and framing to analyse how programming is positioned, regulated and enacted within the technology subject and the wider school context.

Our exploratory, descriptive study focuses on teachers' reflections on their classroom practices. Previous research has highlighted the importance of teachers having a deep understanding of subject content to teach effectively within a specific discipline (Kaya et al., 2022). Based on this, we decided to collaborate with experienced teachers who taught programming before its mandatory inclusion in the curriculum. We regard these teachers as experts, and by exploring their experiences and perceptions, we believe we can capture the most relevant ideas on how to teach programming. By investigating these aspects, we hope to identify the necessary components for integrating programming education effectively into school curricula.

The overarching research question for this study is as follows:

How do teachers of the technology subject perceive what students in grades 4–6 should learn about programming?

More specifically, we seek to understand this research question by examining how teachers reflect on their ePCK - that is, how they implement teaching and assessment practices for programming in the classroom.

Background

Pedagogical Content Knowledge

Pedagogical content knowledge (PCK), introduced by Shulman in the 1980s (Shulman, 1987, 2013), is a framework for understanding teachers' ability to teach a specific subject area. It highlights that classroom practice depends on both pedagogical competence and subject expertise, as well as how teachers apply this knowledge. PCK develops through experience and is shaped by individual and contextual factors (Hubbard, 2018). The framework has been particularly influential in science education and has been continuously refined to also to be used in technology education (eg. Doyle et al., 2019; Hume et al., 2019).

Efforts have been made to develop and update the framework by engaging active researchers in discussions about the framework and its applicability in different summits (Carlston et al., 2019; Gess-Newsome, 2015). The latest model, published in 2019 (Carlson et al., 2019) is called the RCM model which comprises three realms: personal PCK (pPCK), an individual teacher's knowledge and beliefs about teaching specific content; enacted PCK (ePCK), this knowledge in action; and collective PCK (cPCK), the shared professional knowledge found in curricula, research, and teaching communities.

In this study, we focus on how teachers reflect on their classroom practices—descriptions of their ePCK—and how they draw on their pPCK in specific situations. By comparing multiple teachers' accounts, we aim to contribute to understanding their cPCK. Additionally, by asking teachers to relate their reflections to national guidance documents (Swedish National Agency for Education, 2017), we explore how such documents shape the way pPCK is enacted in teaching.

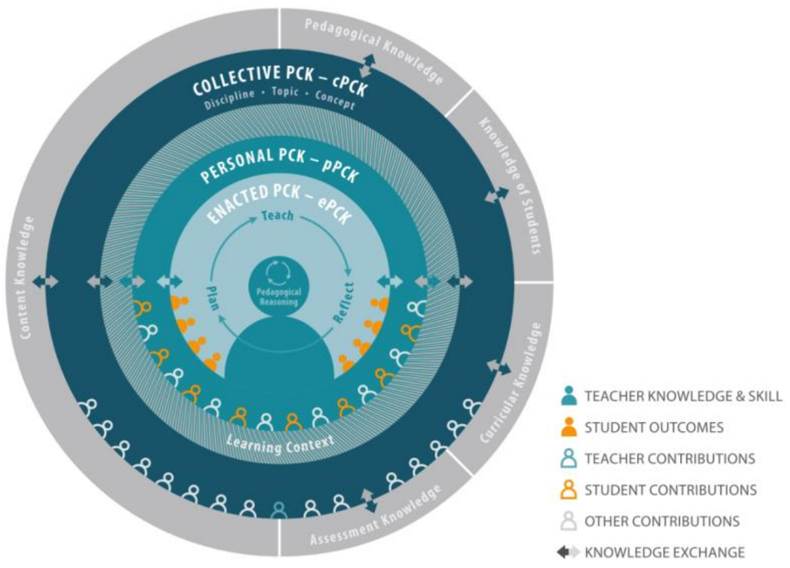

Figure 1. The refined consensus model of PCK (Carlson et al., 2019, p. 84)

Description of the Integration of New Subject Content in the Curriculum

In 2017, programming was introduced into the Swedish compulsory school curriculum as part of the national digitalisation strategy (Nordén et al., 2017). Rather than being defined as a separate subject, it was embedded within existing ones, including technology.

To understand how this new content is valued, organised, and integrated, we draw on Basil Bernstein's theoretical framework. Bernstein (2000) examined how new knowledge domains are transformed when incorporated into curricula and how their status and boundaries are negotiated. His concepts therefore offer a useful foundation for analysing how programming is positioned, classified, and integrated within the technology subject.

Building on Bernstein's work, Ashwin (2009) illustrates how classification helps illuminate how new knowledge is treated in relation to existing disciplinary structures. Bernstein's construct of classification refers to the boundaries between categories of knowledge and can be understood as follows:

1. Strong classification: In this scenario, the new knowledge area is classified as strong, meaning it is included in the curriculum as a standalone content area independent of existing disciplines.
2. Weak classification: The new knowledge area may be integrated as a subset of a well-defined discipline, resulting in weak classification as part of a broader discipline.
3. Non-disciplinary classification: The knowledge area may be regarded as non-disciplinary, serving as a competency that can be developed through project-based courses focused on various knowledge areas. This classification is inherently weak.

Looking at how programming is classified in higher education, we can see several different examples of how it aligns with Bernstein's three classification categories. Programming is clearly strongly classified when it exists as its own discipline in higher education, as seen in educational programs such as software engineering. Conversely, it can also be weakly classified when treated as a tool for solving problems within other knowledge areas. Finally, programming may equally fit with a non-disciplinary classification, whereby students are expected to develop programming skills through project-based courses without being offered foundational courses in the subject.

How programming is classified influences how different stakeholders, policymakers, curriculum designers, teachers, and students, treat and value the content. Understanding these dynamics helps reveal potential tensions and can guide efforts such as teacher training, professional development, and other interventions to support successful integration into school education.

Assessment Practices

Various methods exist to assess individual learning and foster understanding. Formative and summative assessments serve distinct but complementary purposes in education. Formative assessment is an ongoing process that provides real-time feedback to students and teachers, allowing for adjustments in teaching and learning strategies to enhance understanding and performance (Black & Wiliam, 2009). It is primarily diagnostic and developmental, helping to identify areas of strength and weakness, involving teachers as well as learners in the process with the purpose of adapting what happens next to better meet learners' needs based on evidence of learning. Feedback is an integral part of formative assessment, and it needs to be forward-looking providing descriptions on where to go next. Kluger and DeNisi's (1996) review of effective feedback suggests that what seem to matter most is how the learner responds to it.

Summative assessment, on the other hand, evaluates student learning at the end of an instructional period, providing a comprehensive measure of achievement against predefined standards. In Sweden, it is the teachers' responsibility to award the grades to students. It ensures that the learning objectives stipulated in the curriculum have been met (Swedish National Agency for Education, 2011).

By continuing exploring ways to improve the quality of assessment, some researchers suggest that collaboration among teachers is a promising approach (e.g. Harrison, 2009; Hartell, 2015). Collaborative assessment, in which teachers collaborate to construct and assess students' tasks, as proposed by Allal (2013), has been found to enhance both teachers' performance and assessment practices (Thornberg & Jönsson, 2015).

Programming assessment

Assessing programming in compulsory school remains a major challenge. Formative assessment practices such as feedback can support students' motivation and learning (Azmi et al., 2017). Rubrics are frequently recommended for programming assessment (Björklund & Nordlöf, 2023; Brennan & Resnick, 2012) as they help teachers evaluate diverse student solutions (Jönsson & Svingby, 2007; Kilday et al., 2012). However, premade rubrics may cause misalignment with curriculum intentions if teachers adapt them too freely (Sadler, 2009).

In Sweden, summative assessment in the technology subject relies on teachers' own judgments, as no national tests exist for this subject. Such assessments are usually based on classroom activities, with tests being infrequent and of varying quality (Hartell et al., 2018). When it comes to programming, some instruments have been developed. For example, Mannila et al. (2020) created a test to identify students' understanding and misconceptions, but these do not assess the quality of students' self-created programs.

Research shows that programming assessment often occurs ad hoc and focuses mainly on whether a program works, rather than on conceptual understanding or problem-solving processes (Dagienė et al., 2023). Tools such as ProMAT (Hartmann et al., 2022) demonstrate that conceptual knowledge can be assessed, but these tools remain tied to specific environments like Scratch and xLogo. This reflects a broader issue: much programming research relies on Scratch, limiting generalizability (Zhang & Nouri, 2019). Thus, researchers argue for assessment approaches that are conceptually grounded rather than language-specific.

Teachers' assessment practices vary widely. Neutens et al. (2022) found that functionality is often prioritized, while abstraction and testing receive less attention. Swedish teachers similarly emphasize either product-oriented criteria (functional programs) or process-oriented criteria (reasoning, persistence), though the ability to explain code is valued across contexts (Björklund & Nordlöf, 2023). Overall, assessment remains highly individual and teacher-dependent (Dagienė et al., 2023; Hartmann et al., 2022; Zhang & Nouri, 2019).

Although computational thinking is internationally influential (Zhang & Nouri, 2019), the concept is not defined in Swedish policy documents. The curriculum mentions the related notion of “datalogiskt tänkande” but does not specify its meaning or how it should be interpreted in teaching (Swedish National Agency for Education, 2022b). This study therefore focuses on assessing students’ programming knowledge and conceptual understanding as framed in the Swedish curriculum, where programming encompasses problem-solving, creativity, control and regulation, simulation, and democratic dimensions (Swedish National Agency for Education, 2022b).

Steering Documents Guiding Swedish Middle-school Teachers in Programming Education

The Swedish national curriculum consists of three parts: core values and mission, overarching goals and guidelines, and subject syllabuses outlining objectives, core content, and grading criteria (Swedish National Agency for Education, 2022a). These directives serve as the foundation for teachers’ planning, instruction, and assessment but do not contain specifics on how to enact in classroom practices. Instead, teachers are trusted to adapt teaching to their local context. Neither do the Swedish curricula specify detailed content. However, recent national discussions have highlighted a move towards a more knowledge-rich curriculum, inspired by international developments such as in England, emphasising stronger subject content and cognitive foundations for learning (SOU 2025:19, 2025).

To support teachers, the National Agency for Education provides subject-specific materials, including guidance on digital inclusion and clarifications for interpreting the technology syllabus (Swedish National Agency for Education, 2022b, 2022c, 2022d). While these resources offer some support for understanding grading criteria, they still leave considerable room for interpretation, granting teachers substantial autonomy in their professional judgement.

Programming in the Swedish Curriculum

Programming as content has been integrated into both the mathematics and technology syllabuses in Swedish schools since 2018 (Nordén et al., 2017). In the technology subject, the overarching goal is for students to develop an understanding of technological development and its historical context while fostering technological awareness. This knowledge equips students to critically engage with technology, relate its use to issues such as sustainable development, and apply it responsibly. Within this framework, programming is intended to be integrated.

For years 4-6, the technology syllabus specifies three core content areas:

- Technology, people, society and the environment
  Students study technological systems (e.g., water, sewage, recycling) and consider their impacts and trade-offs.
- Technological solutions
  Students learn about components and systems—such as bicycles or electrical circuits—and how these interact to achieve functionality.
- Working methods for developing technological solutions
  Students engage in need identification, proposing ideas, designing, constructing, testing, and documenting solutions using sketches, models, or written descriptions. Programming is included here as a method for controlling constructions or objects (Swedish National Agency for Education, 2022a).

Grading Criteria for Technology (Grade 6)

The grading criteria for technology, like other subjects, are divided into three levels: E, C and A. These correspond to the Swedish national grading scale (Swedish National Agency for Education, 2022a, pp. 277-278), where E = pass, C = good and A = excellent. For example, to achieve an A, the student is expected to perform as follows

> *The pupil gives examples of technological solutions and describes, in a well-developed manner, some of their advantages and disadvantages for the individual and the environment and how they have changed over time.*
>
> *The pupil examines technological solutions and describes, in a well-developed manner, how some components work together to achieve purpose and function.*
>
> *The pupil performs simple technology development and design work in a well-organized manner. In the work, the pupil formulates and chooses alternative courses of action that lead to progress. The pupil creates documentation that sets out the intention of the solution well.*

However, the criteria do not explicitly mention programming in any of the criteria available for E, C, and A. Despite its omission from the text on grading procedures, programming is considered an integral part of the process in technology development and construction work. Therefore, it may be suggested that the third grading criterion should be applied to assess students' programming tasks. This lack of guidance in terms of assessing programming in technology also applies to mathematics.

In mathematics, programming is introduced through visual programming environments to support logical thinking and problem-solving (Swedish National Agency for Education, 2022a). The support material in mathematics offers further guidance on integrating programming, emphasizing the development and application of algorithms in mathematical contexts, such as calculating averages or creating rule-based movements (Swedish National Agency for Education, 2022c). While programming in mathematics focuses on logical and problem-solving skills in technology, programming is approached within a broader context, connecting it to technological systems and their role in society.

There is no doubt that the recent introduction of programming to Swedish schools has posed challenges for teachers in terms of assessment (Vinnervik, 2021), which confirms even earlier results from Hartell (2012) showing that technology teachers are often left to their own devices, and their teaching practices rely solely on their own experiences.

Method

This study is situated within an interpretivist paradigm, assuming that teachers' knowledge and practices are shaped by their experiences, perceptions, and interactions with curricular and institutional structures. Our epistemological stance is constructivist, recognising that understanding is co-constructed through teachers' reflections on their programming instruction. While the PCK framework guides our focus on enacted knowledge, Bernstein's concepts of classification and framing serve as lenses for analysing how teachers interpret and enact the curriculum in their contexts. Together, these perspectives support an exploratory, descriptive approach aimed at deepening understanding of how programming is taught in practice.

Participants

The empirical data are based on interviews and survey responses from seven teachers who consented to being part of the study. All possess prior teaching experience of teaching 10–12-year-olds in programming in technology education, both before and after the official/explicit introduction of programming into the Swedish Curriculum in 2018. Hence, here we consider them as experienced in teaching programming in years 4–6. Their backgrounds show a variety of educational experiences, and all but one has a teacher certificate. Please note that all names are fictious.

Table 1. Background information regarding the participants

Informant	Higher education in programming
Camilla	7.5 ECTS*
Henric	None, self-taught and short course
Irene	None, self-taught
Jack**	36 ECTS in games programming
Kate	Short courses
Liam	Self-taught and short course
Magnus ***	Short courses

* European Credit Transfer System
** No teacher certificates
*** Former teacher, now teacher trainer at a science center

Survey

The initial data for this study were derived from a survey in which participants were asked to underline the parts of the grading criteria in the Swedish technology syllabus they considered relevant for assessing programming skills, as outlined in the central content "Control of the pupils' own constructions or other objects with programming" (Swedish National Agency for Education, 2022a, p. 275).

Follow-up interviews were conducted to gain an even deeper understanding of the participants' views regarding assessment.

Interviews

Seven semi-structured interviews were conducted (Bryman, 2014) by the first author via an online conference system. The interviews were recorded and transcribed verbatim. The interviews aimed to capture multifaceted answers about programming assessment in the technology subject, allowing the participants to articulate their perspectives, methods, and experiences.

Analysis

The data analysis proceeded in three phases. First, we examined teachers' pPCK by examining how the participating teachers interpreted the use of assessment criteria in the technology subject to evaluate students' programming skills. This was done by looking at the text segments they had underlined and by analyzing the interviews. Second, we investigated their ePCK by asking the participants about how they practice their pPCK. Third, we explored the potential for cPCK by comparing the interviews during an open-coded analysis in which we aimed to identify emerging themes. This third process revealed two distinct thematic areas: Programming is the focus vs. programming is the means and Product is the focus vs. process is the focus. These themes provided a structured lens for detecting and understanding how the teachers viewed programming and how its role in education varied. Finally, the identified themes were compared with the existing literature to identify relevant frameworks for deeper analysis. Bernstein's framework, as developed by Ashwin (2009), was particularly useful for categorizing the themes and understanding differences in teachers' assessment practices. Based on the frameworks, a matrix was developed to map the teachers' varied positions along the extremes of the two thematic areas.

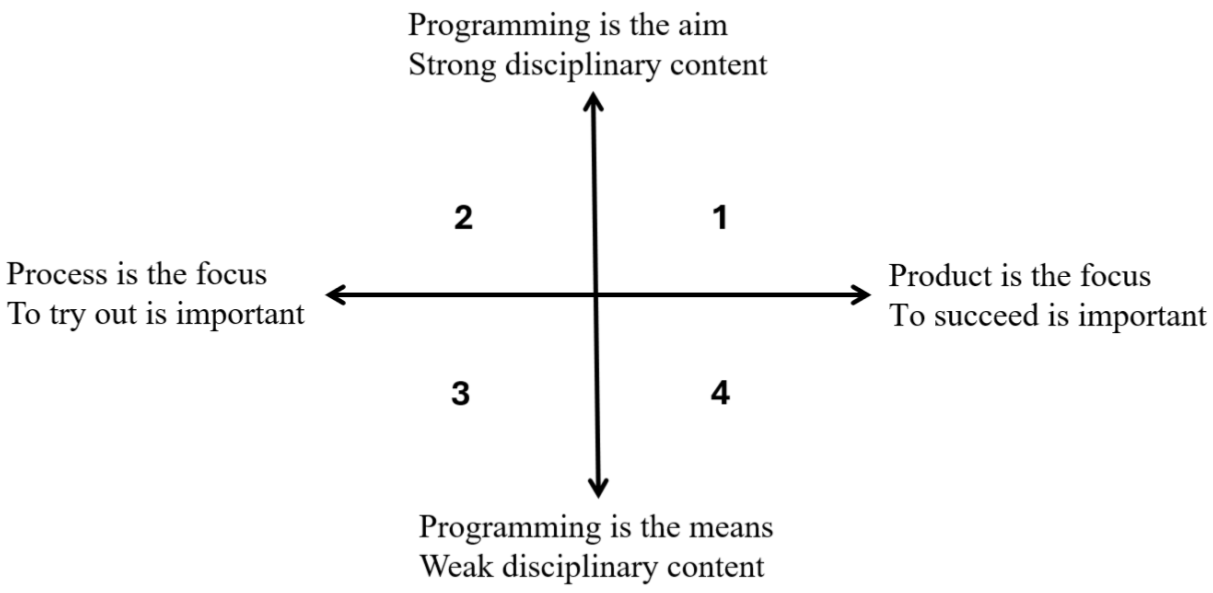

Figure 2. Matrix mapping teachers' various positions in the two thematic areas

The authors compared and discussed the findings and analysis process to reach a consensus.

Results

Teachers' pPCK for Assessing Programming Skills

The participants were asked to underline which grading criteria could be used to assess programming (see example in Figure 3). Liam and Camilla underlined all or almost all the grading criteria. Irene underlined only small portions of the text from all three grading criteria, signalling that not all parts could be used. Kate underlined snippets of the first and third grading criteria and underlined the second grading criterion in full. Magnus, Henric, and Jack all underlined the second and third parts of the grading criteria, while excluding the first grading criteria. In summary, the participants did not agree upon how and if grading criteria one and three could be used, but they all agreed upon the relevance of using grading criteria two for assessing programming skills.

Participant	1st Grading criteria	2nd Grading criteria	3rd Grading criteria
Camilla	The pupil gives examples of technological solutions and describes, in a well-developed manner, some of their advantages and disadvantages for the individual and the environment and how they have changed over time.	The pupil examines technological solutions and describes, in a well-developed manner, how some components work together to achieve purpose and function.	The pupil performs simple technology development and design work in a well-organized manner. In the work, the pupil formulates and chooses alternative courses of action that lead to progress. The pupil creates documentation that sets out the intention of the solution well.

Figure 3. Illustrative example of one participant's underlined parts of the grading criteria considered relevant for assessing computer programming skills (Camilla)

Description of Teachers' ePCK when Assessing Programming

The participants' underlined sections of the grading criteria served as a foundation for the subsequent interviews focusing on assessment. They were asked to further explain how they interpreted the grading criteria to be suitable or not for assessing programming tasks in technology in their daily practice.

The pupil gives examples of technological solutions and describes, in a well-developed manner, some of their advantages and disadvantages for the individual and the environment and how they have changed over time.

The participants shared their thoughts on this grading criterion. Liam noted that some aspects of the criterion are well-suited for evaluating programming projects. He referred to the emphasis on technical solutions and the ability to investigate and describe them. Liam stressed that students' enthusiasm and their ability to articulate their project's goals and functionalities play an important role in the assessment of their programming projects. Liam discussed the relevance of this criterion for his robot war competition, in which students build and program their own robots to compete against each other on a large round mat. The goal is to push the opposing robot off the mat, and the last robot remaining on the mat wins. To participate, students must program their robots to use sensors that detect when they reach the edge of the mat, prompting the robot to reverse to avoid falling off. Liam clearly saw how he could use the first grading criterion to assess programming exercises, explaining, "The technical solutions aspect aligns perfectly with our robot war and classic car projects. These projects inherently involve controlling objects through programming, and it's crucial for students to demonstrate a well-developed understanding of how these components interact. This is essential, as it reflects their ability to build a strong offense and defence in our competitive events. Students often realize this during the competitions when they evaluate their work. Some may say, 'I should have tried something different.' It becomes clear to them why they are not performing at a certain level."

Camilla mentioned that when her students design smartwatches or MP3 players, she can assess with the first grading criterion, as the students reflect on various choices concerning technical solutions. This includes considering their impact on individuals, society, and the environment, as well as recognizing the historical evolution of technology. She explained that using prebuilt blocks results in a pass grade, but when students incorporate their own variables, this raises their level of achievement. Students are given the opportunity to present and discuss the smartwatches or MP3 players they have programmed, which helps Camilla assess them against the first grading criterion. During these discussions, students reflect on the advantages and disadvantages of having a single device that performs multiple functions. In this way, she leverages what the students have created as a foundation for deeper discussions.

The pupil examines technological solutions and describes, in a well-developed manner, how some components work together to achieve purpose and function.

When discussing this grading criterion, the participants consistently emphasized that programming is highly relevant for assessing technological understanding in programming. They particularly highlighted the connections between components within a program and the interactions between an artifact and its corresponding software. Several informants provided evidence supporting this perspective. Jack simply stated that students explore how the different parts in a program come together, while Kate underscored the importance of students understanding the synergy between programming and artifact construction, particularly in the context of LEGO robotics and Micro:bit. Irene believed that she could assess whether the students could describe, in a well-developed manner, how various components interact to achieve specific purposes and functions through programming with Micro:bit. She continued by explaining that this involves connecting different LEDs, making them blink, and lighting up in a certain order.

The pupil performs simple technology development and design work in a well-organized manner. In the work, the pupil formulates and chooses alternative courses of action that lead to progress. The pupil creates documentation that sets out the intention of the solution well.

When addressing this grading criterion, Camilla said that her students formulate and choose action alternatives that advance their programming work. She also mentioned that students sometimes require assistance with individual steps, but she encourages them to seek help and learn through collaboration. Henric described his perspective on how he views a well-crafted technology development project by a student compared to a simpler approach:

> *"Some students tend to rush their work, thinking they're finished in just 10 minutes. I encourage them to spend the next two lessons adding extra features to their games to make them unique. I provide examples, but some find it dull and just change the ball's colour and declare it 'done.' However, with one full lesson and a three-quarter lesson remaining, it's crucial to use the available time wisely. /…/ I think the most important thing is that the students dare to start." (Camilla).*

In addition to students initiating and developing their own ideas, Henric emphasized the importance of simplifying and adapting advanced concepts. He highlighted that students should not only generate ideas for improvement but also demonstrate the ability to refine and adapt complex concepts to make them more accessible. Furthermore, he underscored the value of students' vision and their capacity to identify opportunities. Henric also pointed out that one of the key attributes of programming is that it enables students to articulate and, at times, even present their vision.

To assess the final aspect of the third criterion - students' ability to document their work - Henric evaluates how they write documentation to explain their project's process and functionality. This includes writing game instructions when sharing a game created in Scratch. By allowing students to read each other's instructions and test play their games, they become more aware of how to write clear and effective instructions, ultimately improving their ability to structure and optimize their code.

The third criterion gives the opportunity to assess a technological development process in which students are expected to begin with a need, investigate, propose solutions, construct, and test, using sketches and physical as well as digital models. Jack described this, stating that through the developing phases in technology development and construction work that his students undertake, they get to work with the entire chain. Most of his students' work in the technology subject is centred on programming projects, where they apply their programming skills to develop solutions to various technical challenges.

Regarding the last part of the third criterion, which involves students creating documentation of their work, Liam emphasized its importance. Students take pride in showcasing their work, and Liam takes photos and videos and showcases these for the parents. Students also express a desire to preserve their robot creations for the spring open house, but this may not always be feasible due to other classes needing the material. Consequently, documentation of their work becomes vital for the benefit of others. Additionally, this approach allows students to document their work independently using their personal mobile devices.

In addition to their reflections on the individual grading criteria, the participants expressed more general views on the assessment of students' programming projects. For example, Henric noted that "programming itself isn't a significant part of the assessment," and he mentioned that the curriculum does not specify assessment criteria for programming. However, he believed that the iterative process of refinement should be considered in the evaluation, as it encourages students to experiment and learn through trial and error. Furthermore, Kate's students are in the fifth grade and are not subject to formal grading in terms of summative grades, which begin in year 6 (12-year-olds). Consequently, she does not need to assess them as rigorously as she would if they were in the sixth grade, where grading is mandatory. As a result, her evaluation is more of a pass or not pass nature, based on whether the student has fulfilled the task or not.

On the other hand, Irene, who also teaches fifth-grade students, evaluates them according to the three criteria, and does not use the binary distinction of *pass* or *not pass*.

Exploring the Participants' cPCK

In the thematic analysis of the interviews, two distinct thematic areas that all the teachers discussed and related to were identified. The first concerns how teachers perceive programming as a subject discipline. Specifically, we noted differences in whether the teacher viewed programming as an independent subject with its own terminology and language or as a tool to be used in the context of other subjects. From the literature, we reference Bernstein's pedagogical device approach, as introduced earlier, which describes how a new subject area can be integrated into schools and their curricula.

The second thematic area concerns assessment practices. We observed clear differences between the teachers who emphasized the final product and those who focused more on the process. When compared to the literature, we see that a focus on the product resembles summative assessment, while a focus on the process is more akin to formative assessment (see Assessment Practices).

We viewed the teachers' different responses in those areas as points along a continuum, where one end represents one extreme in the thematic area, and the other end represents the opposite extreme. The teachers who discussed programming as if students needed to prepare for a technical university program were placed at one end of the continuum for the first theme. In contrast, the teachers who expressed that they teach, use, and assess programming as a tool that students need to master to succeed in other subject disciplines were placed at the other end. The teachers who viewed the product (i.e., the program that students created) as the most important aspect to assess were placed at one end of the continuum for the second theme. Conversely, the teachers who believed that the process by which students engaged in programming was the most crucial aspect of assessment were placed at the other end.

Through this analysis, we identified four distinct personas, each situated in a quadrant of a matrix. Each teacher is placed in one of the quadrants. All these personas were visible in this study and encapsulate the diversity of pedagogical approaches among technology teachers in relation to programming education.

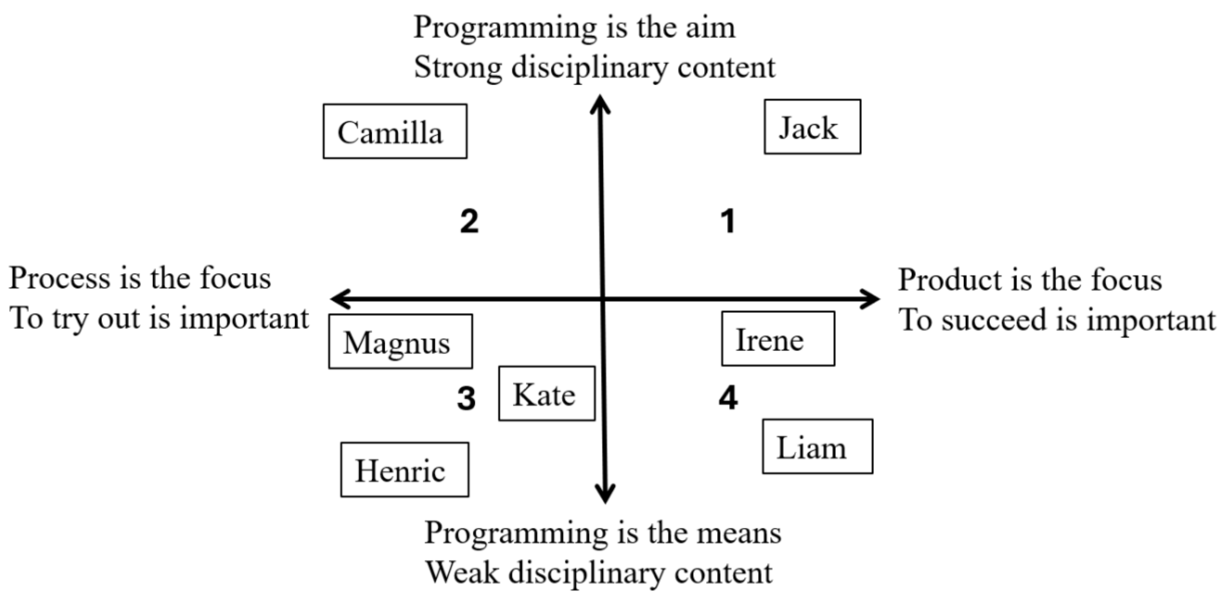

Figure 4. Visualization of how we describe the four distinct personas identified in the study, which depict how technology teachers approach programming instruction

Persona 1: Strong Disciplinary Content, Product Focus

In quadrant one, the teacher views knowledge of programming as a subject area of computer science as a goal of their teaching. The teacher also sees it as important for the student to successfully complete their task and thus be able to master the entire process and present a well-functioning product. When assessing programming, the outcome is important, meaning that the quality of the finished program is considered.

Teacher Jack described how a student who has succeeded/done well and is considered to be in the upper right corner of the quadrant behaves:

> *"... I have this code, and, in this way, it works ... if the student can explain their own solution, they usually can also explain another solution and understand why it works" (Jack).*

According to the same practice, a student close to the origin in the matrix would show rather poor knowledge in the discipline of computer science and would not have succeeded in creating a well-functioning product.

Persona 2: Strong Disciplinary Content, Process Focus

In this quadrant, the teacher focuses on the student learning the basics of programming and thus considers the student's computer science knowledge. However, the focus is not primarily on how the final product works; instead, it is more important that the student has tried different solutions and can demonstrate mastery of coding at various levels.

Teacher Camilla provided an example of how she assesses her students during programming tasks:

> *"... then I can guide them a bit to try that first, because I know there is a greater chance of success ... then I can facilitate, but they have still shown that they have some sort of idea. Now I want this to happen ..." (provide context to make it clear that they are actually discussing the theoretical subject matter) (Camilla).*

This quote also shows that this assessment practice is more formative than summative, as it is very important for Camilla to coach the students during the process. A student who receives a high grade can demonstrate that they have good knowledge of computer science and are willing to try several different methods and solutions.

Persona 3: Weak Disciplinary Content, Process Focus

A teacher whose practice falls into this quadrant focuses on student testing and practice. It is important for the student to use the equipment and the programming environment provided. The process is crucial; thus, it is important for the student to dare to test how coding works. How is assessment done here?

Henric explains how he assesses his students:

> *"... Students try and retry. This idea that you get to try and then you sit down and think about what went wrong and do it again" (Henric).*

Henric also circulates through the classroom, and if students need help, which is quite common, as they often get stuck on some details, he seizes the opportunity to look at the students' code. Additionally, he play-tests their games and asks the students about their programs.

Another example comes from Kate, who emphasized that the actual code and coding proficiency are not her primary focus in assessment. Instead, she evaluates students based on their holistic development, problem-solving skills, ability to enhance their projects, and overall growth and creativity.

A student who receives high grades and is in the lower left corner dares to test and is engaged in and enthusiastic about the task, while a student who does not receive such good grades is not active in the class. Programming and coding are used as a means of activity. This assessment is mainly formative and takes place in the classroom during programming activities.

Persona 4: Weak Disciplinary Content, Product Focus

The fourth quadrant represents the teachers who assess the student's final product. The most important thing is not that the program has good quality coding, but that it truly does what it is intended to do. This assessment practice has a summative focus, and during the class, the teacher tries to support students in achieving a functioning product.

Liam expresses his assessment practice as follows:

> *"... a lot of what fits in is when we actually build and create something, not just clicking in code.org. Then, I would feel that we haven't done enough" (Liam).*

Irene also expresses this with the following statement:

> *"It's important for students to understand how to connect these components and explain why they've done it in a particular sequence." (Irene)*

A student who receives a high grade has managed to get a program to work so that it can control, for example, their own construction. How this has been accomplished is not so important. A student who fails is considered to have failed if they do not produce a functioning product. Even if they have knowledge of coding and have made attempts and experiments, this is not considered in the assessment.

Discussion

Through the PCK framework, we recognize that a teacher's subject knowledge, pedagogical approaches, and understanding of curriculum guidelines and the school context all play a crucial role in how a subject is taught.

In this study, we interviewed seven teachers with expertise in programming to explore how their perspectives on teaching programming in the classroom (ePCK) have evolved over their years of experience. While many aspects of their teaching practices are similar—for instance, all the participating teachers align their instruction with the technology subject curriculum and assessment criteria—their insights showcased the multifaceted nature of assessment, highlighting creativity, problem-solving, and collaboration as pivotal factors in evaluating students' programming achievements. Our findings also revealed that, despite their expertise, they expressed uncertainty about what and how to teach and assess programming. This became particularly evident when examining how they assess students' programming competencies in practice.

The results section first presents teachers' personal PCK (pPCK), followed by a reflection on their enacted PCK (ePCK), aiming to capture how programming instruction and assessment are conducted in the classroom.

It is important to note that we did not directly observe teachers' ePCK in practice; rather, our findings are based on their self-reported experiences of their ePCK.

In the subsequent section, where we discuss the four different personas, we highlight the variations in their teaching approaches. Our analysis suggests that, despite their competence and experience, these teachers demonstrate significant differences in their instructional methods, values, and objectives for programming education. This variation indicates that there is little to no shared collective pedagogical content knowledge (cPCK) among them. By applying Bernstein's theoretical framework (Ashwin, 2009) alongside our understanding of formative and summative assessment (Black & Wiliam, 2009; Swedish National Agency for Education, 2011), we can interpret these differences more clearly. This, in turn, allows us to reflect on the underlying reasons why teachers hold such diverse perspectives on how programming should be taught.

Weak Versus Strong Subject Content

As mentioned, incorporating new subject content into school curricula is not easy. We observe that teachers perceive programming as anything from a relatively weak subject area to well-established and robust disciplinary content. Digitalization is a societal change of unprecedented magnitude, which means there is no prior experience to draw upon in the face of this paradigm shift. By introducing programming as a relatively weak subject content, used merely as a tool within various school subjects, rather than as a discipline that needs to be thoroughly learned, both theoretically and practically, there is a significant risk. Schools may fail to achieve their purpose of providing students with equal opportunities for future career choices. This approach could potentially undermine the development of essential programming skills that are crucial for students' success in technology-related fields. We perceive a risk that students who do not learn the fundamentals of programming from home or are self-taught through their extracurricular activities will find it much harder to both progress in a profession where programming knowledge is a prerequisite and make critical democratic societal decisions. We believe that a deeper understanding of programming and coding is necessary to comprehend today's society and societal development.

Formative Versus Summative Assessment

We observed that formative and summative assessment in programming education often translate into a focus on either the process or the product. Formative assessment emphasizes the learning journey, in which teachers provide feedback on students' ability to understand, find, and apply algorithms to develop a functional program. In contrast, summative assessment evaluates the final product, assessing how well the completed program solves the intended problem (Bjursten et al., 2022).

In this study, the participating teachers demonstrated distinct approaches to formative and summative assessment. Ideally, a balanced assessment strategy incorporating both perspectives would provide a more comprehensive evaluation of students' programming skills. Achieving this balance likely requires strong subject knowledge and a clear understanding of the skills students should acquire in programming. This aligns with the findings of Björklund and Nordlöf (2023), who note that teachers tend to emphasize either the process or the product when assessing programming.

Moreover, the broader assessment culture within the technology subject in schools may influence how teachers approach assessment in programming. Understanding these contextual factors could provide valuable insights into the variations in assessment practices observed in this study.

We identified four distinct teacher profiles or personas, suggesting that students' experiences and knowledge acquisition in programming education vary significantly, depending on their teacher's persona. We argue that it is crucial to reflect on which persona educators should ideally align with when teaching programming in the technology subject for grades 4–6. Such reflection could inform the development of professional development programs tailored to support technology teachers in enhancing their instructional approaches to programming.

What, then, do we believe is the reason behind these relatively large differences?

Diverse Teacher Backgrounds

One of the teachers, representing persona one, Jack, has a background as a software developer before transitioning to teaching technology. As a result, Jack possesses a strong knowledge of computer science. It is unsurprising that this teacher has adopted an approach that emphasizes the importance of students learning the fundamentals of programming as a subject area. It is also understandable that this teacher believes that students can learn both the basics of programming and produce a well-functioning product. Furthermore, we observe that a teacher representing persona two, Camilla, who has a high level of knowledge of computer science and programming, also described her teaching practice as one in which students are encouraged to learn and demonstrate their understanding of the basics of programming. In contrast, the other teachers, lacking the same depth of subject knowledge but having taken short courses in programming, described their teaching practice in quadrants three or four. Thus, they do not prioritize ensuring that students acquire the foundational knowledge needed for learning programming later in their educational journey.

Unclear Policy Documents

Another reason for the disparate views among teachers regarding their practices may be the lack of clarity in the current policy documents. For the technology subject, the curriculum strictly states that students should use programming to control construction. Neither the curriculum nor the supplementary material for technology specifies that students should learn programming (Swedish National Agency for Education, 2022a, 2022d). This ambiguity could be interpreted as suggesting that the curriculum implies that teaching practice should align with persona four, wherein students should be able to use some form of coding to develop a solution without specifying how this should be done. In contrast, the mathematics curriculum describes how aspects of coding practice should be taught, leading teachers who teach both mathematics and technology to prefer quadrant two, whereby it is essential for students to test their knowledge acquired in mathematics in a different context (Swedish National Agency for Education, 2022a, 2022c). Unclear steering documents run counter to the principles of a knowledge-rich curriculum, which is desirable for deep thinking and learning complex skills. In such curricula, clear expectations about what students should learn are crucial for ensuring equitable teaching (Vanhees et al., 2025). This literature also highlights the importance of coherence, both vertical and horizontal: that is, how knowledge for understanding programming is built progressively over time, and how different subjects contribute to a coherent and cumulative understanding of programming.

Limited Programming Competence Among Teachers

We know that approximately 70% of teachers teaching technology express a need for professional development (Swedish National Agency for Education, 2022e). This likely indicates a lack of knowledge about how to program, making it difficult for them to move their practice to quadrants one or two. This may also be the case for our participants, even though they were selected for this study. Through this study, we have reason to believe that a relatively large group of teachers in Sweden find themselves in the teaching practice described in quadrant three, where students have the opportunity to test programming and where curiosity and engagement are the focus of assessment.

Vulnerability in Technology Education

When it comes to technology education, due to the subject's relatively small size, there is often only one technology teacher within a larger teaching environment (Bjursten, Nilsson & Jonsson, 2023). As a result, technology teachers frequently find themselves isolated in their specific practice, with limited opportunities to engage in the collaborative approaches that are commonly advocated for developing a shared teaching culture within a school. This pattern was identified by Hartell (2015) and is, unfortunately, reaffirmed a decade later in recent national data (IVA, 2025). According to the IVA report, many technology teachers still lack fundamental prerequisites for high-quality teaching, including adequate material resources, reasonable group sizes, and sufficient planning time, and they also report limited support from school leadership (IVA, 2025, p. 7).

Conclusion

This study investigated teachers' personal pedagogical content knowledge (pPCK) and their reflections on their enacted pedagogical content knowledge (ePCK) in the context of programming instruction within the technology subject. By conducting a survey, followed by interviews with experienced teachers, we explored how programming is assessed and interpreted within the existing curriculum.

The findings reveal notable variations in how teachers describe their assessment practices, illustrating diverse interpretations of how programming can be assessed. This variation highlights a broader discussion on how collective PCK (cPCK) in programming should be established and integrated in primary education (years 4–6).

Further thematic analysis identified two theme areas influencing teachers' assessment practices: the extent to which they prioritize computer programming as a process for problem-solving versus a product as an end goal and their expectations regarding students' subject content knowledge. The analysis is presented in a matrix with four quadrants representing four personas. Across the study, teachers' assessment practices spanned all four personas: 1) strong disciplinary content and product focus; 2) strong disciplinary content and process focus; 3) weak disciplinary content and process focus; and 4) weak disciplinary content and product focus. This matrix illustrates the different pedagogical approaches taken by teachers, resulting in varying student experiences of programming education.

A critical concern arising from this study is whether Swedish students are equally prepared to engage in a digitalized society. The findings suggest that there are discrepancies in programming education, meaning that students may not receive equal opportunities to develop the necessary digital competencies. Several factors may contribute to this inequality. First, programming remains a relatively new subject in Swedish primary education, and curriculum documents lack clear guidance on how it should be implemented. Second, teachers' backgrounds seem to significantly influence their assessment practices—those with prior experience in programming tend to approach both instruction and assessment differently from those without such expertise. These combined factors indicate that cPCK for programming education has yet to be established, which may have implications for the equity and consistency of digital education in Sweden.

There are several possibilities for strengthening programming as a subject area in schools. One approach is, of course, to ensure that programming is included as a strong subject area in the form of a separate subject in which students learn to program and then use this knowledge in other school subjects. Another approach is to ensure that well-defined subject modules exist in various school subjects. These could be integrated into subjects such as mathematics and technology, as is done today. If these subject modules were well linked with a progressive approach between them, relatively strong subject content could be achieved.

Ultimately, for programming education to contribute to digital equity, it is crucial to reconsider how teachers are trained in programming. A more interdisciplinary approach, integrating both technology and mathematics, could better equip teachers to provide high-quality programming instruction while ensuring equal learning opportunities for all students. Without such measures, the risk remains that students' access to digital competencies will continue to be shaped by individual teachers' interpretations and prior experiences rather than by a standardized and equitable national curriculum.

References

Allal, L. (2013). Teachers' professional judgement in assessment: a cognitive act and a socially situated practice. *Assessment in Education: Principles, Policy & Practice*, *20*(1), 20-34. https://doi.org/10.1080/0969594X.2012.736364

Ashwin, P. (2009). Analysing teaching-learning interactions in higher education. Continuum Publishing Corporation.

Azmi, N. A., Mohd-Yusof, K., & Phang, F. A. (2017). Impact of Effective Assessment towards Students' Motivation in Computer Programming Course. 2017 7th World Engineering Education Forum (WEEF), Kuala Lumpur, Malaysia.

Bernstein, B. (2000). Pedagogy, symbolic control, and identity: Theory, research, critique (Vol. 5). Rowman & Littlefield.

Bjursten, E.-L., Hartell, E., & Gumaelius, L. (2022, 7-10 Dec 2022). Assessment Practices in Computer Programming [Conference presentation]. 11th Biennial International Design and Technology Teacher's Association Research Conference (DATTArc). Southern Cross University, Gold Coast Campus, QLD, Australia.

Bjursten, E.-L., Nilsson, T., & Jonsson, G. (2023). Factors influencing Swedish grades 4–6 technology teachers' choice of teaching and learning material in programming education. *International Journal of Technology and Design Education*. https://doi.org/10.1007/s10798-023-09860-8

Björklund, L., & Nordlöf, C. (2023). Product or Process Criteria?: What Teachers Value When Assessing Programming. In J. Hallström & M. J. de Vries (Eds.), *Programming and Computational Thinking in Technology Education:Swedish and International Perspectives*. Brill. https://doi.org/10.1163/9789004687912_016

Black, P., & Wiliam, D. (2009). Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability*, *21*(1), 5-31. https://doi.org/10.1007/s11092-008-9068-5

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.

Bryman, A. (2014). Samhällsvetenskapliga metoder [Social Research Methods] (6th ed.). Liber AB.

Carlson, J., Daehler, K. R., Alonzo, A. C., Barendsen, E., Berry, A., Borowski, A., Carpendale, J., Kam Ho Chan, K., Cooper, R., Friedrichsen, P., Gess-Newsome, J., Henze-Rietveld, I., Hume, A., Kirschner, S., Liepertz, S., Loughran, J., Mavhunga, E., Neumann, K., Nilsson, P., Wilson, C. D. (2019). The refined consensus model of pedagogical content knowledge in science education. In A. Hume, R. Cooper, & A. Borowski (Eds.), *Repositioning pedagogical content knowledge in teachers' knowledge for teaching science* (pp. 77-94). https://doi.org/10.1007/978-981-13-5898-2

Dagienė, V., Gülbahar, Y., Grgurina, N., López-Pernas, S., Saqr, M., Apiola, M., & Stupurienė, G. (2023). Computing Education Research in Schools. In *Past, Present and Future of Computing Education Research: A Global Perspective* (pp. 481-520). Springer.

Doyle, A., Seery, N., Gumaelius, L., Canty, D., & Hartell, E. (2019). Reconceptualising PCK research in D&T education: Proposing a methodological framework to investigate enacted practice. *International Journal of Technology and Design Education*, *29*(3), 473-491. https://doi.org/10.1007/s10798-018-9456-1

European Union. (2006). Recommendation of the European Parliament and of the Council of 18 December 2006 on key competences for lifelong learning. https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2006:394:0010:0018:en:PDF

Hartell, E. (2012). The inefficient loneliness: A descriptive study about the complexity of assessment for learning in primary technology education KTH Royal Institute of Technology.

Hartell, E. (2015). Assidere necesse est: Necessities and complexities regarding teachers' assessment practices, in technology education KTH Royal Institute of Technology.

Hartell, E., Isaksson Persson, H., Bartholomew, S., & Strimel, G. (2018). Investigating the Potential for RGT and ACJ towards deeper insights of Teacher Assessment Practices. The 36th Pupils' Attitudes Towards Technology Conference in Athlone, Ireland. 18–21 June, 2018,

Hartmann, M., Edelsbrunner, P., Hielscher, M., Paparo, G., Honegger, B. D., & Marinus, E. (2022). Programming concepts and misconceptions in grade 5 and 6 children: Developing and testing a new assessment tool. *Atti del 5° Convegno sulle didattiche disciplinari*, 328-333.

Hubbard, A. (2018). Pedagogical content knowledge in computing education: A review of the research literature. *Computer Science Education*, *28*(2), 117-135. https://doi-org/10.1080/08993408.2018.1509580

Hume, A., Cooper, R., & Borowski, A. (2019). *Repositioning pedagogical content knowledge in teachers' knowledge for teaching science*. Springer. https://doi.org/10.1007/978-981-13-5898-2

IVA. (2025). Det bygger på oss- Tekniklärares syn på sina förutsättningar att undervisa i grundskolan [It's all about us – Technology teachers' views on their ability to teach in compulsory school]. https://www.iva.se/contentassets/4391de43acf443bba3ef7cb8ba26f371/iva-rapport-det-bygger-pa-oss-202509.pdf

Jönsson, A., & Svingby, G. (2007). The Use of Scoring Rubrics: Reliability, Validity and Educational Consequences. *Educational Research Review*, *2*(2), 130-144. https://doi.org/10.1016/j.edurev.2007.05.002

Kaya, Z., Kaya, O. N., Aydemir, S., & Ebenezer, J. (2022). Knowledge of student learning difficulties as a plausible conceptual change pathway between content knowledge and pedagogical content knowledge. *Research in Science Education*, *52*(2), 691-723. https://doi.org/10.1007/s11165-020-09971-5

Kilday, C. R., Kinzie, M. B., Mashburn, A. J., & Whittaker, J. V. (2012). Accuracy of teacher judgments of preschoolers' math skills. *Journal of Psychoeducational Assessment*, *30*(2), 148-159. https://doi-org/10.1177/0734282911412722

Kluger, A. N., & DeNisi, A. (1996). The effects of feedback interventions on performance: A historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological bulletin*, *119*(2), 254-284.

Larsson, A. (2023). *Metaphor in Mind: Programming Teachers' Knowledge and Beliefs in Action* Linköping University Electronic Press. https://doi.org/10.3384/978910752381

Mannila, L., Heintz, F., Kjällander, S., & Åkerfeldt, A. (2020). Programming in primary education: Towards a research based assessment framework. Proceedings of the 15th Workshop on Primary and Secondary Computing Education, virtual event Germany.

Neutens, T., Coolsaet, K., & Wyffels, F. (2022). Assessment of code, which aspects do teachers consider and how are they valued? *ACM Transactions on Computing Education (TOCE)*, *22*(4), 1-27.

Nordén, L.-Å., Heintz, F., Mannila, L., Parnes, P., & Regnell, B. (2017). Introducing Programming and Digital Competence in Swedish K–9 Education. In V. Dagienė & A. Hellas (Eds.), *10th International Conference on Informatics in Schools: Situation, Evolution, and Perspective, ISSEP 2017* (pp. 1-12). Springer Nature. https://doi.org/10.1007/978-3-319-71483-7

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry*, *11*(1), 1-17. https://doi-org/10.1080/20004508.2019.1627844

Sadler, D. R. (2009). Indeterminacy in the use of preset criteria for assessment and grading. *Assessment & Evaluation in Higher Education*, *34*(2), 159-179. https://doi-org/10.1080/02602930801956059

Shulman, L. S. (1987). Knowledge and Teaching: Foundations of the New Reform. *Harvard Educational Review*, *57*(1), 1-22. https://doi.org/10.17763/haer.57.1.j463w79r56455411

Shulman, L. S. (2013). Those who Understand: Knowledge Growth in Teaching. *Journal of Education*, *193*(3), 1-11. https://doi.org/10.1177/002205741319300302

SOU 2025:19. (2025). Kunskap för alla – nya läroplaner med fokus på undervisning och lärande [Knowledge for all – new curricula with a focus on teaching and learning]. (SOU 2025:19). Retrieved from https://www.regeringen.se/rattsliga-dokument/statens-offentliga-utredningar/2025/02/sou-202519/

Stigberg, H., & Stigberg, S. (2019). Teaching programming and mathematics in practice: A case study from a Swedish primary school. *Policy Futures in Education*, 1478210319894785. https://doi-org/10.1177/1478210319894785

Swedish National Agency for Education. (2011). *Kunskapsbedömning - Vad, hur och varför? [Knowledge assessment - What, how and why?]*. https://www.skolverket.se/getFile?file=2666

Swedish National Agency for Education. (2017). Läroplan för grundskolan, förskoleklassen och fritidshemmet 2011 Reviderad 2017 [Curriculum for the compulsory school, preschool class and school-age educare 2011 (revised 2017)] https://www.skolverket.se/publikationsserier/styrdokument/2017/laroplan-for-grundskolan-forskoleklassen-och-fritidshemmet-2011-reviderad-2017?id=3813

Swedish National Agency for Education. (2022a). *Curriculum for the compulsory school, preschool class and school-age educare*. Retrieved from https://www.skolverket.se/sok-publikationer/publikationsserier/styrdokument/2024/curriculum-for-compulsory-school-preschool-class-and-school-age-educare---lgr22

Swedish National Agency for Education. (2022b). Få syn på digitaliseringen på grundskolenivå – Ett kommentarmaterial till läroplanerna förförskoleklass, fritidshem och grundskoleutbildning [Recognizing Digitalization at the Compulsory School Level – Commentary on the Curricula for Preschool Class, After-School Center, and Compulsory School Education]. Retrieved from https://www.skolverket.se/getFile?file=11627

Swedish National Agency for Education. (2022c). *Kommentarmaterial till kursplanen i matematik [Supplementary material to the mathematics curriculum]*. Retrieved from https://www.skolverket.se/sok-publikationer/publikationsserier/kommentarmaterial/2022/kommentarmaterial-till-kursplanen-i-matematik---grundskolan

Swedish National Agency for Education. (2022d). *Kommentarmaterial till kursplanen i teknik [Supplementary material to the technology curriculum]*. Retrieved from https://www.skolverket.se/sok-publikationer/publikationsserier/kommentarmaterial/2022/kommentarmaterial-till-kursplanen-i-teknik---grundskolan

Swedish National Agency for Education. (2022e). Skolverkets uppföljning av digitaliseringsstrategin 2021 [The National Agency for Education's follow-up of digitalization strategy] (RAPPORT 2022:4). Statens Skolverk. https://www.skolverket.se/publikationsserier/rapporter/2022/skolverkets-uppfoljning-av-digitaliseringsstrategin-2021

Teknikföretagen. (2020). Framtidsspaning - Så påverkar teknikskiftena behoven av ingenjörskompeten [Anticipating the Future:The Impact of Technological Shifts on Engineering Competence Needs]. https://www.teknikforetagen.se/globalassets/rapporter--publikationer/kompetensforsorjning/framtidsspaning---sa-paverkar-teknikskiftena-behoven-av-ingenjorskompetens.pdf

Thornberg, P., & Jönsson, A. (2015). Sambedömning för ökad likvärdighet? [Co-assessment for greater equity?]. *Educare - Vetenskapliga Skrifter*(2), 179-205. https://conferences.lnu.se/index.php/PFS/article/view/1401

Vanhees, C., Nijlunsing, J., Muijs, D., Crato, N., Wils, M., Wiliam, D., Surma, T., & Kirschner, P. A. (2025). The role of knowledge-rich curricula in promoting deep thinking and complex skill acquisition. *Learning and Individual Differences*, *121*, 102729.

Vinnervik, P. (2021). När lärare formar ett nytt ämnesinnehåll - Intentioner, förutsättningar och utmaningar med att införa programmering i skolan [Teachers as curriculum makers : intentions, settings and challenges associated with integrating programming into schools] [Doctorial dissertation, Umeå Universitet]. http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1589572&dswid=5480

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education*, *141*, 103607.