

Challenging but Full of Opportunities: Teachers' Perspectives on Programming in Primary Schools

Luisa Greifenstein*
luisa.greifenstein@uni-passau.de
University of Passau
Passau, Germany

Isabella Graßl*
isabella.grassl@uni-passau.de
University of Passau
Passau, Germany

Gordon Fraser
gordon.fraser@uni-passau.de
University of Passau
Passau, Germany

ABSTRACT

The widespread establishment of computational thinking in school curricula requires teachers to introduce children to programming already at primary school level. As this is a recent development, primary school teachers may neither be adequately prepared for *how* to best teach programming, nor may they be fully aware *why* they have to do so. In order to gain a better understanding of these questions, we contrast insights taken from practical experiences with the anticipations of teachers in training. By surveying 200 teachers who have taught programming at primary schools and 97 teachers in training, we identify relevant challenges when teaching programming, opportunities that arise when children learn programming, and strategies how to address both of these in practice. While many challenges and opportunities are correctly anticipated, we find several disagreements that can inform revisions of the curricula in teaching studies to better prepare primary school teachers for teaching programming at primary schools.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education; Computing education programs; K-12 education.**

KEYWORDS

Programming education, primary school, teacher survey.

ACM Reference Format:

Luisa Greifenstein, Isabella Graßl, and Gordon Fraser. 2021. Challenging but Full of Opportunities: Teachers' Perspectives on Programming in Primary Schools. In *21st Koli Calling International Conference on Computing Education Research (Koli Calling '21), November 18–21, 2021, Joensuu, Finland*. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3488042.3488048>

1 INTRODUCTION

Programming is increasingly introduced at primary schools around the world [17]. While secondary school teachers may be expected to be adequately educated in their subjects and computing is usually a dedicated subject, the education of primary school teachers who

*Both authors contributed equally to this research.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Koli Calling '21, November 18–21, 2021, Joensuu, Finland

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8488-9/21/11...\$15.00

<https://doi.org/10.1145/3488042.3488048>

usually teach several subjects without specialising on one particularly [30] has been reported to be insufficient [34]. In order to support primary school teachers it is important to improve their education to better prepare them for programming at primary schools, for example by teaching them to program with SCRATCH [12, 32, 40].

However, beyond acquiring basic programming skills, teacher education needs to cover two further important aspects: First, primary school teachers need to be adequately prepared for the challenges they may face in the classroom; in particular, the challenges arising in practice may be different from those anticipated while studying general education theory. Second, primary school teachers also need to be taught why they are teaching programming to children in the first place, so that they can appreciate the opportunities that may arise for the children as a result of being able to program, and can take measures in order to foster these.

We conducted a two phase survey in order to shed light on both of these aspects. First, we surveyed 97 teachers in training asking them what challenges and opportunities they anticipate to encounter in the classroom. Then, we surveyed 200 teachers in practice asking them which challenges and opportunities they actually encountered in practice, and how they deal with the specific challenges and opportunities that teachers in training most frequently anticipate. Overall, the concerns and opportunities perceived by teachers in training represent a basis upon which the content of teacher training can be improved. The differences in the opinions of teachers in training and in practice can further be used to shape the concept of teachers in training towards a more realistic view. Finally, the ideas of experienced international teachers on how to deal with the challenges and opportunities can enrich teacher training by explaining the suggested strategies.

In detail, we aim to answer the following research questions using the two surveys. First, a primary concern is what challenges teachers in practice encountered in primary schools:

RQ1: *What challenges have primary school teachers in practice encountered when teaching programming?*

By contrasting anticipation and reality, we want to inform primary school teacher education about how to counter anticipated challenges as well as how they align with practice. The second research question therefore is as follows:

RQ2: *What challenges do primary school teachers in training anticipate and what solutions do teachers in practice suggest?*

Effective teaching should also foster and reinforce opportunities arising for children from learning to program. We therefore investigate the following research question:

RQ3: *What opportunities have primary school teachers in practice experienced when teaching programming?*

To see how current teacher education raises awareness of these opportunities, and to improve teacher education by informing teachers in training how to support the opportunities they anticipate, we ask the final research question:

RQ4: *What opportunities do primary school teachers in training anticipate and what promotion do teachers in practice suggest?*

Our results show that many of the challenges and opportunities are correctly anticipated by teachers in training, but there are several points of which they are unaware, such as organisational issues at schools, or where they overestimate importance, such as the complexity of programming. These results can be used to improve teacher training with solutions for existing concerns, central challenges, and possibilities to promote existing opportunities.

2 RELATED WORK

Challenges of Teaching Programming. Changes in the curriculum usually go along with issues for different stakeholders [31]. This also applies to computer science education (CSE), as programming has often only recently been introduced at primary schools around the world [17]. Indeed, for some teachers it is difficult to include the mandatory computing program and therefore they rather neglect it [21]. This might be attributed to challenges that teachers face when teaching programming. There are several studies that analyse challenges faced by CS teachers [13, 18, 34, 37, 38]. These focus on the respective national curricula and on different school levels, but some results seem to be shared: One common challenge is the lacking subject knowledge of teachers, which is mentioned even more often by primary school teachers than secondary school teachers [34]. This might be explained by primary school teachers covering several subjects, while secondary school teachers focus on fewer subjects in which they receive more in-depth training.

Opportunities of Teaching Programming. Few studies explicitly address the holistic potential of programming in primary school, and most highlight and analyse only partial aspects. A leading role is taken by the promotion of computational thinking (CT), which is considered to promote three prevalent framings: skill and competence building, creativity and social and ethical aspects [19]. In particular, cognitive skills such as abstract and logical thinking or inhibition control are emphasised [3, 8]. These skills are especially important because students often demonstrate little algorithmic thinking [10, 20]. The potential of programming skills is often highlighted when considering future careers, as programming is not only important in pure software professions, but also in other industries [36], and society in general [35]. Teachers in practice may not be aware of these scientific studies, therefore surveying teachers about their perceptions and experiences is important.

Effects of Teacher Training. In order to prepare teachers to deal with changes in the CSE curriculum, there are different approaches, mainly concerning training of software-based programming (mainly with SCRATCH), followed by robotics, unplugged programming and game-based learning [4]. Regardless of the technology used it is essential that teacher training provides experiences in practicing as well as teaching programming [22]. This can promote pedagogical content knowledge and self-efficacy and change pedagogical beliefs [22]. This is crucial as teachers in training might have obsolete

prior experience from their time as students [11]. To be able to include the beliefs of teachers in training, they have to be identified first. While there is research on challenges perceived by teachers in practice and on the approaches used in teacher training, there is a lack of bridging these two aspects. In this paper, we combine the opinions of teachers in training and teachers in practice on challenges and opportunities when teaching programming.

3 METHOD

3.1 Study Design

In order to answer the research questions (cf. Section 1), we conducted two surveys: One with teachers in training (TT), and one with teachers in practice (TP), informed by the results of the former. In both surveys, in addition to certain questions given below, all participants were asked about their demographic data, their previous programming experience and their attitudes towards teaching programming in primary school.

3.1.1 Survey 1: Teachers in Training. In the first survey, we asked primary school teachers *in training* to answer two open questions:

- (1) *“What challenges do you see regarding teaching programming in primary schools?”*
- (2) *“What opportunities do you see regarding programming in primary schools?”*

3.1.2 Survey 2: Teachers in Practice. In the second survey, we asked international primary school teachers *in practice* several questions. The first half of the questions dealt with challenges, the second half with opportunities. The first question was the same general question on perceived challenges as in the first survey:

- (1) *“What challenges do you see regarding teaching programming in primary schools?”*

After that, we asked participants to rank the ten challenges that were most frequently mentioned by the primary school TT:

- (2) *“Please rank the following challenges regarding their relevance perceived by you.”*

Then, they were asked to pick the three challenges they perceive as most relevant. For each of these three, they had to answer:

- (3) *“How would you decrease or master these challenges?”*

Finally, there is ongoing research on automated analysis tools to support teachers in addressing the challenges they encounter, but we do not anticipate that such tools are in widespread use yet. We therefore questioned the TP whether they believe automated analysis tools can be useful in practice:

- (4) *“Automatic analysis tools can support teachers with giving feedback to their students. (LitterBox, available at <http://scratch-litterbox.org/>, is an example for an automatic code analysis tool which can find recurring bug patterns in Scratch programs.)”*

They had to state their agreement with this statement using a 5-point Likert scale and explain their rating.

The second half of the survey focused on opportunities:

- (5) *“What opportunities do you see regarding programming in primary schools?”*

The participants were asked to rank the ten opportunities that were most frequently mentioned by the primary school TT:

- (6) *“Please rank the following opportunities regarding their relevance perceived by you.”*

For the three opportunities they perceive as most relevant they then had to answer:

(7) “How would you support or improve these opportunities?”

A much discussed opportunity of programming education is the promotion of girls in computer science. In particular, a matter of debate is the question whether this opportunity can be supported with gender-homogeneous programming classes. Since we do not expect gender-homogeneous classes to be commonplace in primary school practice, we explicitly asked teachers about their opinion:

(8) “Gender-homogeneous programming classes help to improve the opportunity of encouraging girls.”

Again they had to state their agreement on a 5-point Likert scale and explain their rating.

3.2 Participants

3.2.1 Survey 1: Teachers in Training. We implemented this survey into a course on primary mathematics teacher education at the University of Passau in January/February 2021. Participation was voluntary and of 242 TT signed up in the course, 97 (91.8 % female, 8.2 % male) between 18 and 45 years (average 20.37) participated.

To assess the knowledge of the teachers, we asked them about their self-experience in programming. Two-thirds of the TT (67 %) stated that they had not taken any programming lessons themselves at school, while the remaining third had (33 %). To learn about teachers’ mental attitudes regarding the introduction of first programming concepts in order to identify any prior biases, we asked in a 5-point Likert scale whether it is useful to introduce programming in primary school. Among the TT, 45.3 % strongly agreed or somewhat agreed, while 37.1 % were neutral.

3.2.2 Survey 2: Teachers in Practice. To elicit responses for the second survey, we used the Prolific platform¹. As a prescreening filter we restricted participants to teachers working in primary/secondary (K-12) education, with a minimum approval rate of 90, which is the percentage of studies for which a participant has been approved. Using these filters, 1,377 of 147,942 individuals who had been active in Prolific for the past 90 days matched our profile. In order to find out if they had already taught programming in a primary school, we designed a prescreening study with the single question of whether they had already taught programming in primary school, and distributed it to these 1,377 people with a limit of 400 participants. Of these 400, we were able to evaluate 397 responses, 251 of which answered our question in the affirmative and were thus eligible for the main study. The compensation was based on an average wage of £7.50 per hour. The main study was sent to the 251 people from the prescreening study in June/July 2021, achieving a total of 209 respondents. Participants were compensated with £2.50 (based on an average hourly wage of £10.54). We excluded 9 participants because they did not answer all questions.

The 200 (79 % female, 21 % male) experienced international primary school TP were between 20 and 63 years of age (average 36.54). A large majority of the TP currently reside in the UK (70 %), 17 % in the rest of Europe, and 13 % in the rest of the world.

To better assess the knowledge of the teachers, we asked them about their self-experience in programming and their previous

teaching in primary schools. Around half of the TP (53 %) stated that they do not have any prior programming experience. Those who program themselves do this in SCRATCH [29] (25.5 %), Java (21.5 %) or Python (17.5 %). In the classroom, most TP use SCRATCH (65.5 %). PURPLE MASH² is also popular with a third (31.5 %) and exactly a quarter use *unplugged* programming [5]. 19.5 % of the TP also use CODE.ORG. Nearly half (43.5 %) of TP teach programming for 1 hour in primary grades, and 27.5 % teach it for only 30 minutes per week. For most, programming classes are compulsory (70.5 %). Compared to the TT, the TP are more strongly in favour of introducing first programming concepts: 91 % agreed or strongly agreed that it is useful to teach programming and 4 % were neutral.

3.3 Data Analysis

The open questions provided us with qualitative data on which we applied thematic analysis [6]: For each question, we first collected themes, then counted them and in a final step again related them to the original data and our research questions. To ensure inter-rater reliability two raters classified the first 20 statements regarding each open question and agreed on a coding scheme. Each rater then classified half of all statements. Additionally, 40 statements per question were rated by both raters to measure the inter-rater agreement, which is good at $K = 0.69$.³ To answer RQ1 (challenges) and RQ3 (opportunities) we consider the percentage of TP who mentioned the respective code at least once. To answer RQ2 and RQ4 we consider three aspects: First, we compare the percentage of TT who mentioned the respective code with the percentage of TP. We measure statistical differences using a Wilcoxon Rank Sum test with $\alpha = 0.05$. Second, we rank the ten challenges/opportunities mentioned most frequently by the TT based on the percentage of the TP who included them in the three that they deemed important and provided a response for. Third, for the question on analysis tools and gender homogeneous teaching we consider both, the Likert-scale data and the codes resulting from the responses.

3.4 Threats to Validity

External validity: Survey 1 is based on a self-selecting sample as participation was not required to pass the university course, and all TT were from the same university. For survey 2, a large proportion of the TP are from the UK, which is likely because programming has not yet been introduced in primary schools in many countries and thus experiences may be lacking. However, survey 2 nevertheless represents a broad international spectrum.

Internal validity: The TT were given a shorter questionnaire to complete in their spare time, while the TP were monetarily compensated for the more extensive questionnaire. Two authors independently annotated the data, developed categories, reviewed the coding scheme and discussed the few non-matches in detail.

Construct validity: As the survey relies on the authenticity of the respondents, the subjective impressions and experiences are rather to be seen as a guideline. Several researchers besides the authors independently reviewed the questionnaire to reduce the risk of misinterpreting the questions.

²<https://www.purplemash.com>

³For replications, all data of the study including coding schemes are available at <https://github.com/se2p/study-teacher-chall-ops>.

¹<https://prolific.co>

Table 1: Perceived challenges by TP and TT.

The percentage corresponds to the proportion of TP and TT mentioning the subcategory at least once in the open question.

(Sub-) Category	Themes	% TP	% TT
School		45.0 %	19.6 %
organisational	media equipment, funding, internet connection	45.0 %	19.6 %
Students		41.5 %	63.9 %
cognitive	prior media experience, overwhelming, reasoning, literacy, subject knowledge	24.0 %	47.4 %
affective	interest and motivation	10.0 %	10.3 %
heterogeneity	prior media experience, interest, subject knowledge, gender	6.5 %	17.5 %
metacognitive	distractability, concentration, impatience	6.5 %	9.3 %
young age		3.5 %	2.1 %
Teachers		36.5 %	43.3 %
cognitive	subject knowledge, overwhelming, prior media experience	24.5 %	16.5 %
didactic	individual support, child-friendly implementation, prevention of distraction	11.0 %	26.8 %
affective	self-efficacy	7.5 %	11.3 %
Programming		21.5 %	30.9 %
complexity	programming language, technical terms, abstract, relation to life	13.0 %	28.9 %
problems	debugging	9.0 %	2.1 %
Government		17.0 %	13.4 %
organisational	time, curriculum	17.0 %	13.4 %
Parents		4.0 %	11.3 %
organisational	media equipment	3.0 %	6.2 %
affective	fear and criticism	1.0 %	5.2 %

4 RESULTS

4.1 RQ1: Challenges Experienced by TP

To answer RQ1 we consider the responses to the open question on challenges experienced by the TP (survey 2). Table 1 shows the identified categories with their subcategories and themes.

4.1.1 School. The most frequently named challenge with 45.0 % of TP is concerned with schools being organisationally challenged. In particular, this can be split into three individual challenges captured in this response: (TP 189) “school funding issues, particularly around access to resources or even ensuring a consistent internet connection.”

4.1.2 Students. The second most frequently mentioned category with 41.5 % of TP relates to challenges that primary school students might face when programming. Cognitive issues are mentioned most often and partially explained by a lack of digital literacy: (TP 19) “With the younger students it is difficult for them to learn and implement many instructions because they are often still learning basic computer skills like how to use a mouse [...]”. The use of digital media is also linked to metacognitive challenges: (TP 173) “Children are easily distracted when given an exciting new tool to play with, so keeping them on task is a challenge in itself.” Affective factors of students can be even in conflict with other challenges such as (TP 2) “Keeping the programming simple, but still having interesting outcomes to keep the students engaged.” Moreover, all the mentioned characteristics differ between students which leads to the challenge

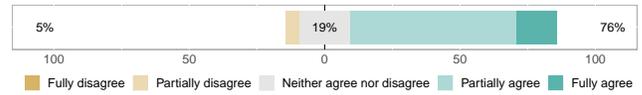


Figure 1: Opinions of TP on analysis tools.

of (TP 76) “trying to teach the whole class at the same time with so many different abilities.” Even if the students might not be heterogeneous in terms of age, (TP 74) “It’s very difficult for younger children to grasp.”

4.1.3 Teachers. Challenges related to teachers are named by 36.5 % of TP. In the context of didactic issues, the most often mentioned challenge regards the individual support: (TP 75) “younger children tend to need a lot of help with this and sometimes it’s hard to get round them all.” Studies on debugging confirm that also high-school teachers struggle with the rush when helping students [23]. Such situations might in turn exacerbate the challenge of low self-efficacy. However, cognitive issues are perceived as the greatest challenge (Table 1) of teachers such as (TP 23) “limited and outdated subject knowledge within teaching staff.” This matches prior research indicating lack of subject knowledge as a major challenge [34, 37, 38].

4.1.4 Programming. A total of 21.5 % of TP name challenges related to programming itself: (TP 123) “At first they have some difficulties understanding the concept, especially that every single step needs to be detailed (as opposed to when you explain something to a human [...])” and (TP 12) “The debugging aspect, which is actually a very powerful learning experience, causes lots of initial impatience!” Thus, programming is related to both cognitive and affective challenges.

4.1.5 Government. A total of 17.0 % of TP see organisational issues regarding time and curricula, and some participants even explicitly connected these related themes: (TP 41) “time allowable in the curriculum for children to understand programming as it’s not an easy task for all to accomplish satisfactorily in the time given.”

4.1.6 Parents. They are hardly mentioned (4.0 % of TP), and mainly regarding media equipment: (TP 114) “Some children in disadvantaged schools dont [sic] have easy access to computer systems at home.”

RQ1 Summary. The main challenges of TP are organisational issues of the school, and cognitive issues of teachers and students.

4.2 RQ 2: Challenges Perceived by TT and Strategies for these from TP

To answer RQ2 we consider the open question on the perceived challenges (survey 1 and 2), as well as the remaining questions on challenges and strategies of survey 2. Table 2 shows the ten most common challenges mentioned by TT, ranked by TP. Table 3 includes the strategies that TP mentioned for the top ten challenges of TT. Fig. 1 shows the rating of analysis tools.

4.2.1 Challenges Perceived by TT. All categories and subcategories are mentioned by both TT and TP (Table 1). However, the distribution differs, which can be explained by significant differences in the subcategories. In the following we discuss these differences.

Students. TT mentioned cognitive challenges significantly more often ($p < 0.001$). This indicates that students are less cognitively

Table 2: Top ten challenges of TT ranked by TP.
The % TP corresponds to the proportion of TP ranking the respective challenge among the three most relevant challenges.

Nr.	Category	Challenge	% TP
1	School	There might be a lack of technical equipment at primary schools.	69.5 %
2	Teachers	Teachers might be challenged with cognitive issues (e.g. because of a lack of subject knowledge).	53.5 %
3	Government	Institutions might be organizationally challenged (e.g. because of an overloaded curriculum and programming being time consuming).	35.5 %
4	Teachers	Teachers might not know which didactic considerations and methods they should use to teach programming (e.g. how to master looking after each student and their individual programming issues).	30.5 %
5	Teachers	Teachers might have negative attitudes (e.g. because of their self-concept towards programming).	21.0 %
6	Programming	Programming might be a (too) complex topic for primary schools.	20.5 %
7	Students	Students might be cognitively overwhelmed and might have insufficient prior knowledge regarding digital literacy.	20.0 %
8	Students	Students might have problems because of metacognitive issues (e.g. by being easily distracted).	11.5 %
9	Parents	Parents might have negative attitudes or there might be a lack of technical equipment at home.	10.5 %
10	Students	There might be differences between the students (resulting e.g. from girls having different tendencies or preferences than boys).	3.5 %

Table 3: Strategies of TP for the challenges of TT.
The absolute values refer to all mentions and the percentage to the proportion of TP mentioning the subcategory at least once.

Category	Subcategory	1	2	3	4	5	6	7	8	9	10	Σ	% TP
Teachers	participate in training	0	101	1	45	0	29	1	3	0	0	180	68.0 %
	teaching	12	2	10	11	0	1	29	26	21	5	117	42.5 %
	content of training	0	26	1	14	0	23	0	0	0	0	64	23.5 %
	cooperate with others	4	6	2	7	0	6	3	3	0	0	31	15.5 %
	schedule time	0	0	11	1	0	0	9	4	2	1	28	14.0 %
School	use existing material	0	9	3	5	0	4	0	0	0	0	21	10.5 %
	media equipment	96	0	0	2	4	0	0	1	0	0	103	43.5 %
	course after school	0	0	7	0	2	0	0	0	0	0	9	4.5 %
	joint approach	0	0	5	2	0	0	1	0	0	0	8	4.0 %
	responsible teacher	1	2	3	0	0	0	0	0	0	0	6	3.0 %
Government	more funds for CSE	51	0	2	0	1	0	0	0	0	0	54	26.5 %
	reorganise curricula	0	0	29	1	0	0	7	11	0	0	48	23.0 %
Parents	content	0	0	0	0	10	0	3	0	0	0	13	6.0 %
	meetings	0	1	1	0	6	0	1	0	0	0	9	4.0 %
	support	0	0	0	0	6	0	0	0	0	0	6	3.0 %
Society	rethink sociologically	0	0	0	0	0	0	0	0	0	2	4	1.0 %
Σ		164	147	75	88	29	63	54	48	23	8	1400	

overwhelmed and have more sufficient media experience and subject knowledge than TT assume. Interestingly, the theme “reasoning” is mentioned only by TP which is a more general cognitive skill and associated with CT [33]. A similar picture emerges regarding the challenge of heterogeneity: TT consider this aspect significantly more often as challenging than TP ($p = 0.002$). This might also be attributed to only TT mentioning gender differences as an issue.

Teachers. TT mentioned challenges regarding the teachers slightly more often (Table 1). TT perceive the didactic issues as significantly more challenging than TP ($p < 0.001$). Consequently, TP do not seem to be overwhelmed to choose an appropriate teaching approach. Therefore, TT can benefit from the strategies and working methods used by TP that are explained in Sections 4.2 and 4.4.

Programming. TT put a different focus (Table 1): TT mentioned the complexity of programming significantly more often ($p < 0.001$) and TP the debugging process when teaching programming ($p = 0.026$). This is why debugging should be supported in the classroom [23] or within teacher training [15].

School. TT seem to underestimate the organisational issues of schools as TP mentioned them significantly more often ($p < 0.001$). This can be attributed to TT not having considered funding and internet connection but only the issue of media equipment.

Government. Regarding the government, there are no significant differences: Both TT and TP mentioned curriculum and time issues.

Parents. Although parents are mentioned least often by both groups, TT considered them almost three times as often (with 11.3 % vs. 4.0 %) and particularly overestimated affective issues ($p = 0.027$).

4.2.2 Strategies. The ten most common challenges mentioned by TT were ranked regarding their relevance by the TP (Table 2). For the three challenges that the TP ranked most relevant, they were asked to describe a strategy (Table 3). In the following, we explain for each challenge what strategies are suggested.

Challenge 1: Lack of equipment at schools. For the most relevant challenge (Table 2) both school and governmental strategies are considered: The government should provide more funds for CSE and furthermore, TP suggest different possibilities to get media equipment beyond simply buying it such as: (TP 136) “Schools should also hold fund raising events—the children will often be happy to take part in events that will raise money for technology or ‘toys’ for them to use.” A different approach considers dividing students into groups which leads to a reduced need of technical equipment.

Challenge 2: Cognitive issues of teachers. In order to counteract the lacking subject knowledge, TP suggest participating in training that addresses content knowledge but also affective factors. Indeed, training with 68 % of TP is the most often mentioned strategy across all challenges (Table 3). According to TP, training should be more often, high-quality, for free, mandatory and prior to teaching.

Challenge 3: Organisationally challenged institutions. The lack of time implies that curricula should be reorganised by (TP 26) “Introducin [sic] digital thinking in earlier years” and (TP 147) “Decrease or make other priorities less time consuming.” Further ideas are listed in Table 3. In this context, the strategy of the teaching approach mostly deals with teaching interdisciplinary as (TP 201) “Coding could be linked to Literacy/Numeracy learning in some way—that would benefit motivation in those subjects and offer time to do it.”

Challenge 4: Didactic issues of teachers. TP suggest to participate in teacher training which focuses on pedagogical content knowledge rather than content knowledge. Besides training, within

the teaching approach simply trying out different methods and (TP 12) “peer mentoring among teaching staff” are suggested.

Challenge 5: Affective issues of teachers. Interestingly, as for cognitive issues, teacher training should both address content knowledge and affective factors such as attitudes, motivation and confidence. This might be explained by well-informed teachers being confident: (TP 38) “[...] an expert came to talk to the staff about ‘Scratch’ and showed us how to do a variety of things. This certainly helped boost the confidence of the less technology literate members of staff.”

Challenge 6: Complexity of programming. This might be reduced by the teaching approach such as (TP 160) “Making sure we can simplify the concepts to relate to real life examples first, before going heavy on the programming concepts.” This matches the strategy of “contextualisation” found by Sentance and Csizmadia [34].

Challenge 7: Cognitive issues of students. All student-related challenges (7, 8 and 10) are expected to be improved by the the teaching approach of considering individual factors such as interest or strengths, e.g., by differentiating learning material. Cognitive issues in particular are often stated to decrease when the students get the chance of expanding their prior knowledge: (TP 148) “We would not introduce them straight to programming; instead, teach them how to better use technical equipment first.” Programming with hardware is addressed here, the explained previous promotion of digital literacy however is not relevant for programming unplugged [5].

Challenge 8: Metacognitive issues of students. Besides considering individual factors, metacognitive issues can be counteracted with increasing the students’ engagement by, e.g., a (TP 160) “distract free environment.” and (TP 198) “interactive activities [sic].”

Challenge 9: Affective and organisational issues of parents. Comprehensibly, this challenge is connected to all strategies regarding parents (Table 3). TP explained that (TP 15) “for the negative attitudes of parents, they can be invited for a day to see how their children work and how much fun they have” or to (TP 139) “create easy to follow guides to allow the parents to acquire knowledge.”

Challenge 10: Heterogeneity of students. Besides considering individual factors when teaching, there might be a need for rethinking sociological views: (TP 4) “Girs [sic] are usually raised for household chores and not more mental issues. this is a societal problem.”

4.2.3 Automated Analysis Tools. There exist several tools that analyse block-based programs [2]. Recent approaches for SCRATCH programs deal with, e.g., next-step hints [26] or code perfumes [25]. However, TP do not appear to be well aware of the potential of automated tools to support programming education, as 29.5 % of TP explicitly stated that they had no experience with such tools.

However, Fig. 1 shows that TP like the idea and consider automated tools potentially useful in terms of giving feedback. Giving feedback can be located in challenge 4 (didactic issues of teachers) as individual support is mentioned most often here (Table 1). Besides giving and perceiving feedback, debugging, diagnosis, support and assessment are mentioned. This goes along with several explained advantages such as dealing with many students at the same time, feedback being immediate or more confidence of teachers. Consequently, these advantages might also help with cognitive

Table 4: Perceived opportunities of TP and TT.

The percentage corresponds to the proportion of TP and TT mentioning the subcategory at least once in the open question.

(Sub-) Category	Themes	% TP	% TT
Skills acquisition		55.0 %	81.4 %
cognitive	problem solving and logical thinking, linguistic stimulation	15.5 %	9.3 %
holistic	creativity, digital literacy, computational literacy	26.5 %	59.8 %
affective	interest and motivation, reduction of prejudices, fun, self-confidence, talent	23.0 %	33.0 %
metacognitive	self-reliance	1.0 %	6.2 %
Foundation		34.5 %	34.0 %
professional life		15.0 %	7.2 %
general future		7.5 %	18.6 %
secondary school		6.5 %	14.4 %
universal		7.5 %	0.0 %
Cross-curricularity		10.5 %	3.1 %
Society		10.0 %	28.9 %
digitalisation		10.0 %	26.8 %
student’s lifeworld		0.0 %	4.1 %
Early support		7.0 %	29.9 %
Diversity		5.5 %	10.3 %
girls		2.0 %	7.2 %
alignment		3.5 %	2.1 %
girls and boys		0.0 %	2.1 %
Methods		3.5 %	16.5 %
exploring-discovery		2.5 %	7.2 %
playful-child-oriented		1.5 %	6.2 %
variety		0.0 %	3.1 %

and affective issues of teachers and the often described lack of time [34, 37]. (TP 170) “With so many students, and so much to do in the classroom, these tools allow teachers to provide feedback [sic] to their students without needing to spend several hours reading code.” Indeed, automatically generated hints can support teachers with debugging programs if the hints fulfil certain criteria [15].

Some TP reveal scepticism against tools and believe one should rather rely on human feedback: (TP 127) “I don’t think it can give a subjective feedback.” However, the lack of individualisation implies one advantage of computer-based feedback: Tools give objective feedback—independent of, e.g., gender or social background [16].

RQ2 Summary. TT partially consider other challenges than TP, e.g., regarding the difficulty of programming. Generally, TP regard teacher training and the teaching approach as most helpful. Analysis tools are considered useful for giving efficient feedback.

4.3 RQ3: Opportunities Experienced by TP

To answer RQ3 we consider the open question on opportunities experienced by the TP (survey 2). Table 4 shows the seven identified categories of opportunities with their subcategories and themes.

Skills acquisition. The most frequently mentioned opportunity is the acquisition of skills (55.0 %), which is about building new knowledge and abilities, or about repeating and deepening what has been learned. It is categorised into metacognitive, cognitive,

ffective, and holistic factors. According to the TP, digital and computational literacy in particular (6.0 %, 11.5 %) as well as a creative component (8.5 %) are fostered through programming. On a cognitive level, programming is especially important for logical thinking and problem solving (14.0 %), whereas it is striking that a small proportion of TP also see linguistic support (2.0 %). Starting programming at an early age also offers opportunities for affective factors, first and foremost the expression of interest and motivation (8.5 %). In addition to fun (4.5 %), active programming in primary schools can reduce prejudices and fears about programming and computer science in general (8.5 %). Furthermore, self-confidence and independence of the students are promoted (6.0 %).

Foundation. According to more than a third of the TP, programming is an important building block for transferring knowledge for the future (34.5 %). This includes not only skills for the general future life of the students, but also their time at secondary school and later working life: (TP 65) “Programming is becoming a market in terms of employment and I think teaching children it from a young age will help them develop skills which they could take forward into the future careers and can be built upon at secondary schools.”

Cross-curricularity. The opportunity of interdisciplinary learning and teaching while programming is the third most frequently mentioned category (10.5 %), which mainly benefits (TP 123) “maths and informatics” or (TP 162) “programming can be integrated across a range of subject areas, including maths, English, science, geography, history, music and art.”

Society. Similar to interdisciplinarity, programming lessons are seen as an essential opportunity to keep pace with the increasing digitalisation of society (10.0 %): (TP 170) “Programming is the foundational language of technology, which is quickly becoming the foundation of society. I think studnets [sic] understanding basic progroaming [sic] is like undtstanding [sic] the basics of a car.”

Early support. It is particularly early support, beginning in primary school, that 7.0 % of the surveyed teachers see as fundamental: (TP 150) “I think elementary school is the perfect time to give students exposure to the foundations and building blocks of coding. If the seeds can be planted at an early age the [sic] can develop their skills as they progress through school.”

Diversity. Programming might strengthen equal opportunities for all students regardless of gender or background by reducing mutual reservations and possible stereotypes (5.5 %), (TP 44) “especially for girls and those children who struggle or are turned off by core subjects” and (TP 136) “very often, children who are not so academic—and don’t do well in other lessons—can really fly in computing lessons.”

Methods. There are opportunities for different teaching and learning methods (3.5 %), such as (TP 0) “group work and interaction” or (TP 3) “more active learning”.

RQ 3 Summary. The main opportunities of TP are the development of skills on different levels, interdisciplinarity and a foundation for the future. The early and diversity-related promotion as well as the methodical implementation are also relevant.

Table 5: Top ten opportunities of TT ranked by TP.

The % TP corresponds to the proportion of TP ranking the respective challenge among the three most relevant challenges.

Nr.	Category	Opportunity	% TP
1	Skills	Students gain cognitive skills (e.g. problem solving skills).	65.0 %
2	Foundation	Students might acquire knowledge on which they can build on in their future.	54.5 %
3	Skills	Students acquire competencies in terms of digital literacy (e.g. being able to use a computer).	46.5 %
4	Society	Programming and related skills are important in our society (e.g. because of the increasing digitization).	40.0 %
5	Skills	Students acquire positive attitudes towards programming (e.g. increased interest in programming).	26.0 %
6	Skills	Students are encouraged in their creativity (e.g. by creating an own program).	24.0 %
7	Skills	Students acquire competencies in terms of computational literacy (e.g. understanding how algorithms work).	15.5 %
8	Early support	Students are promoted at an early stage.	12.5 %
9	Diversity	Boys and especially girls acquire competencies (e.g. wrt. affective aspects such as motivation).	8.5 %
10	Methods	Teaching programming implies new or varied teaching methods.	7.0 %

4.4 RQ 4: Opportunities Perceived by TT and Strategies for these from TP

To answer RQ4 we consider the perceived opportunities stated by both the TT (survey 1) and TP (survey 2), as well as the remaining questions on opportunities and strategies of survey 2. Table 5 shows the ten most common opportunities mentioned by TT ranked by TP and Table 6 includes the proposed strategies by TP. Figure 2 shows the opinions of TP on gender-homogeneous classes.

4.4.1 Opportunities Perceived by TT. All categories and subcategories are mentioned by both TT and TP (Table 4). However, the distribution differs which can be explained by significant differences in the subcategories. In the following we discuss these differences.

Skills acquisition. In agreement with the TP (Table 4), building new knowledge is the most relevant opportunity (81.4 %) for the TT, although the TT mention it about a quarter more often. Both groups see cognitive skills as an essential opportunity. At the holistic level, the TT focus on digital and computing literacy as core skills acquired through programming, which differs significantly from the TP ($p < 0.001$ and $p < 0.001$, respectively). This points to a misconception of programming [27] as many TT agree that programming will serve (TT 17) “to encourage children at an early age in the proper use of technology, media, and the Internet.”

In creativity, both groups see an equal opportunity. In affective factors, TT identify similar opportunities as TP, but they emphasise increasing student interest and motivation and talent ($p = 0.022$): (TT 26) “Particularly at primary school age, one discovers a number of interests, gifts or talents. Accordingly, programming could also be one of them, which means that I can well imagine that a new field of interest will open up for some students”. TP might not share this view due to their practical experience that students are already very

Table 6: Strategies of TP for the opportunities of TT.

The absolute values refer to all mentions and the percentage to the proportion of TP mentioning the subcategory at least once.

Category	Subcategory	1	2	3	4	5	6	7	8	9	10	Σ	% TP
Society and personal relevance	future opportunities	3	26	10	19	2	0	1	0	1	0	62	28.5 %
	student's lifeworld	17	8	5	13	1	1	2	1	1	1	50	23.5 %
	relevance of prog.	3	9	8	18	3	0	3	0	0	0	44	20.5 %
	practical application	5	10	4	6	1	2	4	0	2	0	34	16.0 %
Methods	universal	20	13	16	0	5	8	8	3	0	4	77	33.0 %
	task variety	13	6	7	4	6	4	2	2	1	1	46	19.5 %
	individual needs	12	2	3	1	2	3	1	0	1	0	25	11.5 %
	programs	4	2	0	0	1	1	2	1	0	0	11	4.5 %
Affective factors	creativity	5	0	0	0	0	21	0	1	1	0	28	14.0 %
	encouragement	3	6	0	3	10	3	1	2	2	0	30	13.5 %
	mindset and inclusion	1	7	2	3	4	1	0	1	7	2	28	13.0 %
	universal	5	1	1	0	12	2	0	1	2	1	25	11.0 %
Organisational factors	alternative thinking	5	1	0	0	1	5	1	1	1	0	15	7.0 %
	requirements	8	12	21	15	4	4	3	1	1	4	73	27.0 %
	inviting experts	0	4	0	4	2	0	1	1	0	0	12	6.0 %
	Programming	8	10	12	9	2	2	5	1	0	0	49	22.5 %
Cross-curricularity	25	5	5	3	2	2	3	1	0	1	47	21.0 %	
Early support	1	5	1	0	3	0	0	13	0	0	23	10.5 %	
Σ		138	127	95	98	61	59	37	30	20	14	1358	

interested in programming and computer science in general, as the survey on the girls' courses (Section 4.4.3) also indicates.

Foundation. Developing skills for the future life is one of the central opportunities for both TT and TP. The TT, in contrast to the TP, place their focus here on competencies that may be important for the general future ($p = 0.005$) as well as for secondary school ($p = 0.026$). Since the teachers already using programming in class have experienced the children's use and opinions, they may already have more concrete ideas about possible career goals.

Early support. Early support is a much more relevant opportunity for beginning teachers than for those already practicing ($p < 0.001$), which might be due to the fact that TP already teach it.

Society. The TT mentioned social relevance about twice as often as the TP ($p < 0.001$). In addition to increasing digitalisation, they also referred to the impact on children's lifeworlds, which the TP did not address at all. TP may not only see the specific lifeworld reference, but also embed it in a larger societal context.

Methods. For the TT, the methodological didactic application is a much more important opportunity than for the TP, especially to provide a (TT 29) "a change from the 'normal' forms of teaching for the children", thus differing in methods ($p = 0.013$) and child-oriented implementation with reducing cognitive load ($p = 0.028$).

Diversity. The TT consider the promotion of diversity as a slightly more relevant opportunity than the TP. They see opportunities especially for girls ($p = 0.026$), whereas the equalisation of all students to one level is seen as similar by both groups.

Cross-curricularity. Interdisciplinarity was mentioned by TT, but significantly less than by TP ($p = 0.028$), which may be related to their lack of connection factors due to little field experience.

4.4.2 Strategies for Opportunities. The ten most common opportunities mentioned by TT were ranked regarding their relevance by the TP (Table 5). For the three opportunities that the TP ranked most relevant, they were asked to describe a strategy. We explain for each opportunity what strategies are suggested (Table 6).

Opportunity 1: Cognitive Skills. The most relevant opportunity, gaining cognitive skills (65.0 %), can be best promoted with a more interdisciplinary curriculum (Table 6): (TP 81) "Linking problem solving with other areas of the curriculum eg maths". These potential competencies in turn affect other subjects: (TP 2) "Learning cognitive skills will help students in other subjects, such as math and english."

A range of methodological and didactic approaches can also be highly supportive for developing cognitive skills (Table 6). On the one hand, the variety of tasks is important; for instance, activities like team work, workshops or homework help to better understand programming. In addition, the potential of programming competitions has often been highlighted. On the other hand, meeting the individual needs of students is an important component. Thereby the focus is on feedback and the application of suitable learning strategies as one TP mentioned: (TP 0) "By differentiating my teaching, e.g., by varying the content and the approaches so as not to exclude [...] the ones who do not find programming extremely related to their needs or learning orientation." In addition, positive reinforcement was often referred to: (TP 114) "Encourage some kind of reward system for digital superstars." Moreover, when the teacher addresses the children's life world, it promotes their cognitive abilities.

Opportunity 2: Foundation. Pointing out and actively discussing the many job and career opportunities that exist in computer science has the most beneficial effect on the second most relevant opportunity, the children's future (54.5 %). This includes explaining and demonstrating all the jobs that use programming (Table 6), and also giving (TP 121) "examples of people who currently work in programming and what their job involves".

Opportunity 3: Holistic Skills. Organisational factors (Table 6) constitute the basis for students' digital literacy which is the third most relevant opportunity (46.5 %). Requirements such as the school's media equipment and a restructuring of the timetable that allows sufficient time for programming lessons play a crucial role. In addition, teacher training can contribute to teaching digital literacy to students. In order to develop digital literacy at all, a variety of methods should be considered, and particularly the basics and craft of programming must be supported: (TP 99) "Have them code".

Opportunity 4: Society and personal relevance. By explaining the relevance of programming and potential job opportunities, the relevance of the subject (40.0 %) is made more accessible: (TP 45) "Presentation of various areas of life where computer skills are important and that without them you cannot function in the real world."

Opportunity 5: Affective Skills. Students' positive attitude towards programming (26.0 %) can be achieved in particular by enthusiastic teachers acting as role models when they (TP 118) "inspire, empower and engage". Moreover, teachers should encourage students by, e.g., (TP 65) "developing motivation, passion, commitment and patience" and (TP 54) "making it a fun activity".

Opportunity 6: Creativity. In order to unleash the potential of creativity (24.0 %), TP suggest that it should be explicitly stimulated by (TP 12) "encourag[ing] students to go 'off piste' and use theor [sic] programming in different ways" or (TP 143) "open ended tasks which enable children to be more creative with programming e.g. design a game with certain criteria rather than program this traffic light".

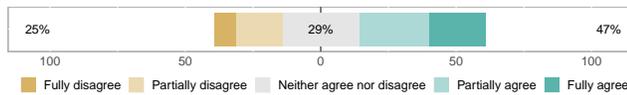


Figure 2: Opinions of TP on gender-homogeneous classes.

Opportunity 7: Holistic skills. Computational literacy can be promoted through different methodological approaches (Table 6), e.g., a gradual and playful implementation was emphasised by the TP: (TP 136) “Begin with simple algorithms/instructions they are already able to understand and follow, such as food recipes or instructions for getting up in the morning. Build on those. Card sorting activities can help support those who struggle with ordering instructions.”

Opportunity 8: Early support. Rather intuitively, early promotion can be met with precisely such an approach, since (TP 62) “the earlier children explore the basics of coding, the more easily they will be able to learn, understand, and apply coding later in life.”

Opportunity 9: Diversity. Diversity can best be promoted by positively changing the attitudes of parents and teachers (Table 6), e.g., by doing more public relations work such as students giving (TP 102) “a presentation about programming, to parents or during an assembly” and by establishing an inclusive climate, where the school can also be a place where equal opportunities are created, since (TP 174) “many [students] do not have access to computers at home. So it is a must to provide the opportunity at school.”

Opportunity 10: Methods. In order to apply the types of teaching that programming education requires, teachers need to be taught how to use (TP 0) “new tools that facilitate both teaching (approaches, load, differentiation of content and process) and learning [...]”

4.4.3 Gender Homogeneous Groups. Both TT (7.2 %) and TP (2.0 %) see the promotion of girls as an opportunity for programming in primary schools, and in general, diversity in computer science has been gaining importance in recent years [1]. One way to get girls interested in programming might be to offer special courses just for girls [41]. Figure 2 shows that the majority of TP consider such courses to be positive or are neutral toward them.

Many teachers report that they experienced few to no affective (7.5 %) and cognitive (3.5 %) differences between the genders in primary school, although there may be preferences thematically [14], as (TP 73) “boys are more interested in ‘boy’ type games and programs. The girls do enjoy minecraft which is more neutral.”

While teachers see it as important to promote girls, they are also skeptical since mixed courses give the opportunity to impact the working environment in the long-term: (TP 189) “In order to break down the stereotyping which exists around IT and other technologically related career paths, I believe that they should be taught in mixed classes where girls are able to show off how skilled they are at coding. By doing so we will have a new generation of men that respect the programming abilities of their opposite gender.”

Gender bias may also be present at the teacher level and should be included: (TP 136) “I think teacher awareness and support is much more relevant. There may be biases towards boys doing better, but that will not always be the case and the teacher needs to push all pupils to the best of their abilities, whilst keeping in mind the biases that exist

in their class or school.” Above all, interaction between the genders should be encouraged: (TP 7) “cooperation instead of antipathy”

Since diversity has several aspects besides gender, some teachers see the bigger picture in pointing out the importance of other underrepresented groups: (TP 44) “In my district, there is a focus on raising the attainment of children from poorer families and this would be a bigger concern than the gender attainment gap.”

Overall, teachers see the support of girls as very relevant, but are ambivalent about whether homogeneous courses are a suitable implementation for this, in-line with current public discourse [1].

RQ 4 Summary. The main opportunities of TT are similar to those of the TP, but TT emphasise early support, society and methods. The most conducive strategies of TP are various methods, interdisciplinarity, and demonstration of the relevance of programming. Many TP are positive about promoting girls through extra courses for girls, but see a danger of further stereotyping.

5 DISCUSSION

Considering challenges in conjunction with opportunities, some overarching misconceptions of TT and strategies of TP are recognisable. For example, TT seem to overestimate the importance of students’ digital literacy. This might be related to even in-service teachers having difficulties regarding terms such as ‘programming’ or ‘computational thinking’ [9, 24]. The surveyed TP, in contrast, focused on more general cognitive skills such as reasoning and problem solving, which are strongly associated with CT [33]. This goes along with the idea that programming should not be taught as an independent skill but rather be used as an approach to promote CT [24]. Teacher training should therefore explain programming as a vehicle to promote ways of thinking rather than detached skills.

Teacher training should provide strategies to use programming to teach computational thinking concepts. When comparing the strategies for the associated challenges and opportunities, the teaching approach is emphasised (Tables 3 and 6). TP suggest, e.g., to consider individual needs but also interdisciplinary teaching, as cross-curricular teaching supports dealing with different challenges and opportunities and moreover, is even an opportunity itself. Likewise, the Computer Science Teachers Association and International Society for Technology in Education (ISTE) points out that CT also needs to be taught across multiple curricular disciplines [28] as one teacher suggests: (TP 119) “Put programming as a stand alone lesson to teach skills but also include [sic] it in other lessons to show how versatile it is and to link it to real life problems.” In particular, programming can refer to selected contents of other subjects and thus establish a synergy of different learning areas. Conceivable learning effects for a simple game in, e.g., SCRATCH would be as follows: Math (coordinate system, angles, mirroring), Languages (syntax, verbal and nonverbal signals, rules), Physics (movement, gravitation, distance and time measurement), Music (gaits, dance, sounds and noises), Art (colours) and general studies (senses, food, clothing, dances, rituals and festivals).

Such a cognitively reduced, playful and interdisciplinary approach is not only useful for children [7], but also for teachers. Especially for the TT, SCRATCH courses at university can reduce fears of contact through self-experience [40] and also convey the competencies that are relevant for their future teaching practice [39].

6 CONCLUSIONS AND FUTURE WORK

In order to bridge the gap between concept and practice and provide a realistic picture of programming education for teachers, we surveyed 200 TP and 97 TT regarding their opinions on the challenges and opportunities of teaching programming in primary schools. Overall, our results show that, despite some challenges, the majority of both TT and TP support the introduction of programming into primary schools and see it as valuable.

TP and TT perceive challenges regarding the school, students, teachers, programming, government and parents, whereas TT set a different focus. To counter challenges, TP consider especially teacher training and the teaching approach as useful. Their anticipation of the help provided by automated analysis tools provides reinforcement for the research, development, and evaluation of such tools. According to TP, programming classes also provide opportunities, especially to develop skills on different levels which children will need for their future life. Similar to the challenges, the TT and TP identify similar opportunities but emphasise them differently. Programming might promote girls into computer science, though different course designs should be reviewed in field studies. The many opportunities will only yield benefits if programming is also embedded in the curriculum, and if the strategies identified in this study are applied in teacher training and evaluated in practice.

ACKNOWLEDGMENTS

This work is supported by the Federal Ministry of Education and Research through project “primary::programming” (01JA2021) as part of the “Qualitätsoffensive Lehrerbildung”, a joint initiative of the Federal Government and the Länder. The authors are responsible for the content of this publication.

REFERENCES

- [1] K. Albusays, P. Bjorn, L. Dabbish, D. Ford, E. Murphy-Hill, A. Serebrenik, and M.-A. Storey. 2021. The diversity crisis in software development. *IEEE Software* 38, 2 (2021), 19–25.
- [2] N. D. C. Alves, C. G. Von Wangenheim, and J. C. Hauck. 2019. Approaches to assess computational thinking competences based on code analysis in K-12 education: A systematic mapping study. *Informatics in Education* 18, 1 (2019), 17.
- [3] B. Arfé, T. Vardanega, and L. Ronconi. 2020. The effects of coding on children’s planning and inhibition skills. *Computers & Education* 148 (2020), 103807.
- [4] M. Ausiku and M. Matthee. 2020. Preparing Primary School Teachers for Teaching Computational Thinking: A Systematic Review. *Learning Technologies and Systems* (2020), 202–213.
- [5] T. Bell, I. Witten, and M. Fellows. 2015. CS Unplugged: An enrichment and extension programme for primary-aged students. (2015).
- [6] M. M. Bergman. 2010. Hermeneutic content analysis: Textual and audiovisual analyses within a mixed methods framework. *SAGE Handbook of Mixed Methods in Social and Behavioral Research*. Thousand Oaks, SAGE (2010), 379–396.
- [7] M. U. Bers, C. González-González, and M. B. Armas-Torres. 2019. Coding as a playground: Promoting positive learning experiences in childhood classrooms. *Computers & Education* 138 (2019), 130–145.
- [8] S. Çiftci and A. Bildiren. 2020. The effect of coding courses on the cognitive abilities and problem-solving skills of preschool children. *Computer science education* 30, 1 (2020), 3–21.
- [9] I. Corradini, M. Lodi, and E. Nardelli. 2018. An investigation of Italian primary school teachers’ view on coding and programming. In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*. Springer, 228–243.
- [10] V. Dagiene, L. Mannila, T. Poranen, L. Rolandsson, and P. Söderhjelm. 2014. Students’ performance on programming-related tasks in an informatics contest in Finland, Sweden and Lithuania. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*. 153–158.
- [11] P. A. Ertmer and A. T. Ottenbreit-Leftwich. 2010. Teacher technology change: How knowledge, confidence, beliefs, and culture intersect. *Journal of research on Technology in Education* 42, 3 (2010), 255–284.
- [12] G. Fesakis and K. Serafeim. 2009. Influence of the familiarization with “scratch” on future teachers’ opinions and attitudes about programming and ICT in education. *Acm SIGCSE Bulletin* 41, 3 (2009), 258–262.
- [13] C. Girvan, C. Conneely, and B. Tangney. 2016. Extending experiential learning in teacher professional development. *TEACH TEACH EDUC* 58 (2016), 129–139.
- [14] I. Graßl, K. Geldreich, and G. Fraser. 2021. Data-driven Analysis of Gender Differences and Similarities in Scratch Programs. In *Proceedings of the 16th Workshop in Primary and Secondary Computing Education (WiPSCE)*. ACM. To appear.
- [15] L. Greifenstein, F. Obermüller, E. Wasmeier, U. Heuer, and G. Fraser. 2021. Effects of Hints on Debugging Scratch Programs: An Empirical Study with Primary School Teachers in Training. In *Proceedings of the 16th Workshop in Primary and Secondary Computing Education (WiPSCE)*. ACM. To appear.
- [16] J. Hattie. 2008. *Visible learning: A synthesis of over 800 meta-analyses relating to achievement*. routledge.
- [17] F. Heintz, L. Mannila, and T. Färnqvist. 2016. A review of models for introducing computational thinking, computer science and computing in K-12 education. In *FIE '16*. 1–9.
- [18] M. Israel, J. N. Pearson, T. Tapia, Q. M. Wherfel, and G. Reese. 2015. Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education* 82 (2015), 263–279.
- [19] Y. Kafai, C. Proctor, and D. Lui. 2020. From theory bias to theory dialogue: embracing cognitive, situated, and critical framings of computational thinking in K-12 CS education. *ACM Inroads* 11, 1 (2020), 44–53.
- [20] T. Koulouri, S. Lauria, and R. D. Macredie. 2014. Teaching introductory programming: A quantitative evaluation of different approaches. *ACM Transactions on Computing Education (TOCE)* 14, 4 (2014), 1–28.
- [21] L. R. Larke. 2019. Agentic neglect: Teachers as gatekeepers of England’s national computing curriculum. *BJET* 50, 3 (2019), 1137–1150.
- [22] S. L. Mason and P. J. Rich. 2019. Preparing elementary school teachers to teach computing, coding, and computational thinking. *Contemporary Issues in Technology and Teacher Education* 19, 4 (2019), 790–824.
- [23] T. Michaeli. 2021. *Debugging im Informatikunterricht*. Ph.D. Dissertation.
- [24] B. Munasinghe, T. Bell, and A. Robins. 2021. Teachers’ understanding of technical terms in a Computational Thinking curriculum. In *ACE*. 106–114.
- [25] F. Obermüller, L. Bloch, L. Greifenstein, U. Heuer, and G. Fraser. 2021. Code Perfumes: Reporting Good Code to Encourage Learners. In *Proceedings of the 16th Workshop in Primary and Secondary Computing Education (WiPSCE)*. ACM. To appear.
- [26] F. Obermüller, U. Heuer, and G. Fraser. 2021. Guiding Next-Step Hint Generation Using Automated Tests. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*. 220–226.
- [27] S. Papadakis, M. Kalogiannakis, and N. Zaranis. 2016. Developing fundamental programming concepts and computational thinking with ScratchJr in preschool education: a case study. *IJMLO* 10, 3 (2016), 187–202.
- [28] B. B. Ray, R. R. Rogers, and M. M. Hocutt. 2020. Perceptions of non-STEM discipline teachers on coding as a teaching and learning tool: what are the possibilities? *Journal of Digital Learning in Teacher Education* 36, 1 (2020), 19–31.
- [29] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, et al. 2009. Scratch: programming for all. *Commun. ACM* 52, 11 (2009), 60–67.
- [30] P. J. Rich, S. F. Browning, M. Perkins, T. Shoop, E. Yoshikawa, and O. M. Belikov. 2019. Coding in K-8: International trends in teaching elementary/primary computing. *TechTrends* 63, 3 (2019), 311–329.
- [31] J. Ryder. 2015. Being professional: accountability and authority in teachers’ responses to science curriculum reform. *Stud Sci Educ* 51, 1 (2015), 87–120.
- [32] J. M. Sáez-López, J. del Olmo-Muñoz, J. A. González-Calero, and R. Cózar-Gutiérrez. 2020. Exploring the Effect of Training in Visual Block Programming for Preservice Teachers. *Multimodal Technologies and Interaction* 4, 3 (2020), 65.
- [33] C. Selby and J. Woollard. 2014. Refining an understanding of computational thinking. (2014).
- [34] S. Sentance and A. Csizmadia. 2017. Computing in the curriculum: Challenges and strategies from a teacher’s perspective. *Educ Inform Tech* 22, 2 (2017), 469–495.
- [35] P. Tuomi, J. Multisilta, P. Saarikoski, and J. Suominen. 2018. Coding skills as a success factor for a society. *Educ Inform Tech* 23, 1 (2018), 419–434.
- [36] A. Vee. 2017. *Coding literacy: How computer programming is changing writing*. Mit Press.
- [37] P. Vinnervik. 2020. Implementing programming in school mathematics and technology: teachers’ intrinsic and extrinsic challenges. *International journal of technology and design education* (2020), 1–30.
- [38] A. Yadav, S. Gretter, S. Hambrusch, and P. Sands. 2016. Expanding computer science education in schools: understanding teacher experiences and challenges. *Computer Science Education* 26, 4 (2016), 235–254.
- [39] S. Yeni, E. Aivaloglou, and F. Hermans. 2020. To Be or Not to Be a Teacher? Exploring CS Students’ Perceptions of a Teaching Career. In *Koli Calling*. 1–11.
- [40] E. Yukselturk and S. Altıok. 2017. An investigation of the effects of programming with Scratch on the preservice IT teachers’ self-efficacy perceptions and attitudes towards computer programming. *BJET* 48, 3 (2017), 789–801.
- [41] Z. Zhan, P. S. Fong, H. Mei, and T. Liang. 2015. Effects of gender grouping on students’ group performance, individual achievements and attitudes in computer-supported collaborative learning. *Comput. Hum. Behav.* 48 (2015), 587–596.