Supporting Non-CS Teachers with Programming Lessons

Svetlana Unkovic[0009−0004−5538−2971] and
Martina Landman[0000−0002−0274−4172]

TU Wien, Karlsplatz 13, Vienna, Austria

Abstract. The introduction of a new compulsory subject in secondary schools and the publication of the syllabus in Austria poses great challenges for many teachers. A crucial point is the fact that the selected teachers often do not have an adequate degree in computer science or similar training. Consequently, the lack of their CS competencies often leads to an insufficient teaching, as there were not specifically trained enough for this subject before its introduction. Our outreach programme "eduLAB" offers an initiative where teachers in schools are supported by student staff in teaching programming. To evaluate this initiative, 28 teachers were asked how our programming courses and offerings could be adapted and expanded to support teachers and also students in the new compulsory subject. This paper analyses the survey results and ideas of what we can do to further support teachers in this situation.

Keywords: programming course · Programming with Processing · outreach to school teachers · CS school curriculum · teacher training.

1 Introduction

In Austria, a rethinking of digital education and computer science education resulted in the replacement of the pre-scheduled exercise by the compulsory subject "Basic Digital Education" (BDE) in 2022, which combines aspects of media education and computer science. One hour per week is dedicated to this subject from the 5^{th} to the 8^{th} grade.

The number of teachers trained in the field is by far not sufficient to fill the open positions, which means that many teachers have to fill in without the proper training and background. We are interested in a first assessment of the teachers' own view on the matter. Based on existing literature [7] we hypothesise that teachers feel insecure about teaching the subject area. Moreover, we aim to answer the following two research questions:

RQ1. How do teachers who teach the compulsory subject "Basic Digital Education" in secondary schools rate their prior programming knowledge?

RQ2. How can our programming course for 9^{th} grade be adapted to make it useful for the compulsory subject and thus support teachers?

J.-P. Pellet and G. Parriaux (Eds.): ISSEP 2023 Local Proceedings, pp. 61–74, 2023.
https://doi.org/10.5281/zenodo.8431918

2 Related Work

The importance of teaching computer science in lower secondary education is being recognized worldwide and included in existing curricula or new school subjects, as well as the early implementation of this curriculum [12]. Introducing programming into curricula is also advocated by official guidelines such as the Informatics Curriculum or the new subject "Basic Digital Education"[2].

One example of a programming initiative is Scratch[1], which provides visual block programming as a learning tool and guided projects as teaching materials. Another example is Code.org[2], offering coding resources, curriculum materials, and professional development programs for teachers. There are many more online initiatives that offer teaching materials as well as teacher training courses (e.g. `coderdojo.com`, `codeclub.org`, `codeweek.eu`, etc.). In contrast, our approach is to support teachers directly in their classroom and to provide additional teaching materials. A good example of a successful introductory programming course is shown by Porter and Simon at UCSD, which focused on the three aspects pair programming, peer instruction and media computation[9].

However, new content in the curriculum leads to new teacher responsibilities. Especially if they lack adequate training for new curriculum content, they may struggle to support their students and feel overwhelmed [7]. A study, which compared teachers' attitudes towards CS skills, discovered a shortage of adequately trained teachers who require assistance in teaching programming skills [13]. The need of better training is also highlighted in [11]. Additionally, the new curriculum seems cryptic according to teachers without proper CS education (see 3.2). The importance of including computer science in national secondary school curricula underline the need of our approach to support untrained teachers in teaching programming.

Furthermore, we used Processing [10], a programming language known for its extensive visual representation capabilities, into our course to enhance the visualization of programming concepts. Starting from our Processing course, originally designed for university students, we adapted and tailored it for upper secondary school classes [5]. Based on teachers' support, we expect to see greater interest in computer science and a better understanding of programming, which can be reinforced through such outreach activities [4].

3 Structure and Content of the Compulsory Subject

The aim of the introduced subject is to provide future adults with early educational and professional opportunities as well as private advantages over so-called "digital illiterates"[6].

[1] `https://scratch.mit.edu/`
[2] `https://code.org/`

3.1 Syllabus

A look at the syllabus reveals that the compulsory subject is built on the five competence areas "orientation", "information", "communication", "production" and "action"[2].

- **Orientation**: "... analysing and reflecting on social aspects of media change and digitalization."
- **Information**: "... dealing responsibly with data, information and information systems."
- **Communication**: "... communicating and cooperating by using information and media systems."
- **Production**: "... creating and publishing digital content, designing algorithms and programming: Decomposing problems, recognising patterns, generalising/abstracting and designing algorithms."
- **Action**: "... assessing offers and possibilities for action in a world shaped by digitalisation and using them responsibly."

In addition, the content is categorized in technological, social, and interactional domains. These perspectives are based on the "Frankfurt Triangle" [1] which purpose is to interdisciplinarily guide and structure educational processes in digital transformation, involving all relevant disciplines.

For the sake of simplicity, they can be described by three questions **(T)** *"How do digital technologies work?"*, **(G)** *"What are the social interactions that result through the use of digital technology?"* and **(I)** *"What are the options for interaction and action for pupils?"*.

In order to close the circle between the introduced model and the five competence areas, we want to show their connection: Each competence area is subordinated to the three presented perspectives *T*, *G* and *I*. It must be added that a competence area does not have to be limited to only one description of a perspective. On the contrary, several descriptions from one perspectives can be found in a competence area. In addition, some competence areas contain areas of application that can be used for teaching, but by far not all areas are supported with these suggestions.

3.2 Remarks and Criticism on the New Syllabus and Realisation

By studying the syllabus closely, including individual discussions with teachers during our (programming) workshops and their feedback on teaching at school, it becomes apparent that the strong generalisation and cryptic description of certain sub-areas *T*, *G* and *I* can quickly lead to perplexity and confusion in the preparation of the lessons. The abstract explanations and lack of examples and descriptions in the fields of application, if they even exist, make designing the teaching units an obstacle.

In addition, the quick introduction of the new compulsory subject leads to open teaching positions which need to be filled. Due to the shortage of teachers non-specialist teachers without training or background in computer science are

therefore obliged to teach this subject. Particularly young teachers are affected, as we have learned from informal conversations among young teachers that they are often presumed by older colleagues having the know-how and experience. Still the repertoire of knowledge in teaching is missing and by many older colleagues not considered. From this, the affected teachers face the problem of not knowing which and how they can adequately convey certain contents in class. Additionally, it has to be taken in account that they often have to become familiar with the contents themselves. This problem is also pointed out in other literature [14]. To support teachers, Austria's Federal Ministry provides offers at the University College of Education and a MOOC "Basic Digital Education"[3] as part of a continuing and further education programme. The question has to be raised whether it is possible to pack an entire university course or a teacher training programme into such a framework. There is as well an offer to an existing university course "Teacher of Basic Digital Education"[3] which corresponds to a duration of two years.

This gives us hope and future outlook that one day qualified persons will fill these places through this university course. Unfortunately, due to shortage of study places in this course and teachers in general, this initiative and approach could turn into a rather unrealistic concept. Nevertheless, our research group wants to support, reach out to those teachers who are lacking in programming, and also provide remedial support in the future.

4 Setting

We are offering a short online programming course for teachers and young students alike, originally targeted at 9th grade students. The course is based on and uses Processing[4] as a programming language. The Austrian curricular guidelines demand, however, that programming be taught in 8th grade (the guidelines regarding programming are shown in table 1).

Level	Excerpts from the syllabus	
8th grade	(T)	"Design and iteratively develop programs that combine control structures,including nested loops and compound conditionals."
	(I)	"cooperate with provided media and software applications in a purposeful and creative way
	(I)	"create simple programs or web applications using appropriate tools to solve a specific problem or accomplish a specific task."

Table 1: Overview of the competence area "Production" from the curriculum of the subject "Basic Digital Education".

[3] https://www.ph-noe.ac.at/de/weiterbildung/hochschullehrgaenge/
[4] https://processing.org/

We have therefore begun to modify our original course for 9th graders to better fit the target age group of 8th graders. Additionally, there is a number of teachers who took the course with or booked it for their students. The course's contents are briefly discussed in the following subsections.

4.1 Programming Crash Course as a Workshop in Schools

Our free programming course focused on "Computer Science", aligning with the syllabus objectives from 9th grade onwards. A year ago, we restructured our previous workshop into a four hour programming "crash course". In this programming workshop, pupils have the opportunity to get a taste of the Java-like programming language Processing. Since the language allows graphic output right from the start, the students can quickly see results and experience their first successes in programming.

The workshop is run by our student staff and tutors in schools and, in exceptional cases, online and is designed for two double sessions[5]. The flexibility makes the workshop very popular, depending on the choice only one double lesson can be used or both. The workshop implementation follows a team-teaching format with the teacher. There is always a short introduction to the programming concepts, followed by a free working session. However, the hands-on session is always longer than the theoretical and practical introduction by the workshop leaders. In the practical section, the students have the opportunity to work in groups and support each other.

Our tasks and materials are accessed via a website link provided by our staff during the first session. In addition to the tasks, the students receive a "cheat sheet" that lists the most important Processing commands, and a student-friendly and adapted script with all important explanations from our MOOC course created for first-year university students [5]. Moreover, we provide solutions on request with uniform commentary notation, as well as a detailed description of the program sequence.

Looking back to the periods between March 2022 and May 2023, our offer was used by five schools, including eight groups with 187 pupils. Seven of these groups were in the 9th grade and one group in the 8th grade, of which five groups of the 9th grade opted for the four hour programming block. In the 8th grade, the workshop leaders noticed that the students needed more time to understand certain programming concepts compared to the 9th graders. Therefore, only the first double session was conducted. Nevertheless, it must be said that starting from one supervised group, it is not possible to draw conclusions about all 8th graders. However, since the subject BDE only lasts until the 8th grade and programming is part of the syllabus, we consider offering a slimmed-down version of the course for the 8th grade or possibly introduce another environment, such as microworlds [8], which would then be prepared on the basis of our programming

[5] The duration of a session corresponds to 50 minutes. CS classes are usually held in double sessions.

crash course from the 9th grade onwards. For this purpose, we conducted a questionnaire to find out the needs of the teachers who teach BDE.

Between December 2022 and May 2023, five teachers, including one outside the country, requested access to the website and to the Moodle course which contains more content, in total 11 lessons [5].

4.2 Comparison of the previous and new version of the course

We noticed that certain contents were not well received by the students, so we skipped contents such as "defining and working with own variables", "compound comparisons in branches" or "arithmetic operations" from our workshop program. Instead, we have included or adapted materials and tasks that focus on understanding the concepts, such as "calling methods", "predefined variables", "loops" and "branches". The changes in the course can be seen in table 2 and 3.

	OLD VERSION	NEW VERSION
access	▪ moodle course with multiple duplicated course ▪ login data needed	▪ website ▪ without login data
	block 1 (two sessions)	
	"basic shapes"	"basic shapes and pre-defined variables"
content	▪ Programming environment - Processing surface - orientation in the sketch window: coordinate system - comments and indentations of the code - difference between setup( ) and draw( ) - first commands with rect( ), ellipse( ) and triangle( ) ▪ Pre-defined variables: - height, width - mouseX, mouseY ▪ Own variables - color	

Table 2: Comparison of the old and new version of the course - Part 1.

5 Data collection and methodology

We have conducted a survey among the first group of teachers who worked with our programming course and teachers who are teaching BDE. Some of

	block 2 (two sessions)	
	"variables and branches"	"pre-defined variables and branches"
content	▪ own variables - color - boolean ▪ simple branches and compounded branches ▪ functions	▪ pre-defined variables ▪ simple branches

changes
▪ deletions in the content ▪ simple explanations and connecting to previous knowledge of the students ▪ from three to two basic tasks in each content block ▪ more creative tasks in the additional task section ▪ adjustments to the scripts ▪ no introductory videos ▪ removed sections like "own variables" and related tasks

Table 3: Comparison of the old and new version of the course - Part 2.

the teachers who participated do not have a formal training in computing as such. In total, we received answers from 28 participants (11 female and 17 male teachers), all of whom teach grades 5 to 12. The survey was mostly multiple-choice-questions (MCQ) or Likert-scale questions (from 1 to 5). The survey's structure is shown in table 4.

Similar research in this area shows that the lack of inclusion of certain CT perspectives in computer science classes is due to a lack of prior programming knowledge and non-existent CT knowledge [14]. Learning motivation and fear of failure are cited as the main reasons and the biggest challenge. With our questionnaire we want to check whether teachers really need support, especially in the area of programming, and if so, which measures need to be set.

The online questionnaire was carried out in spring 2023. We used various channels for dissemination, such as our newsletter or contacting schools that attended our workshops. The questionnaire contains 26 questions that were grouped in five categories. Bifurcations were built into the questionnaire. Some sections could be skipped, depending on whether someone had taken the programming crash course before or not. This way, BDE teachers could be filtered out from CS teachers. However, the focus is on BDE teachers and how they can be guided by our research group in terms of programming. The remaining categories serve the general goal of improving our existing programming course.

Category 1	**Demographic data of the participants**
The gender, professional experience and teaching subjects of the participants were recorded at the beginning of the questionnaire.	
Category 2	**Digital education**
– assessment of the infrastructure at the school – access to learning materials and further and in-service training – reasoning for teaching the subject "Digital Basic	
Category 3a	**Our Outreach programme**
– participation in the offers of the programme (e.g. unplugged workshops)	
Category 4	**Own programming experience**
Objective 1	Prior programming knowledge of the teachers.
– previous knowledge and self-assessment in programming – repertoire of programming languages – motivation for participating in the programming	
Category 5	**Programming course Processing**
Objective	Feedback and attempt at ongoing improvement of materials and implementation.
– preparation of the contents in the first and second programming block – difficulties encountered in programming and programming concepts by learners – suggestions for improvement – continuation with the programming language in the classroom and use of the Moodle course	
Category 3b	**Appeal of the programming Crash Course**
Objective 2	Support BDE teachers through existing programming course (with customisation) or other programming opportunities.
– naming the reasons for participation – desires for other programming languages or microworlds – interest in further education and training	

Table 4: Overview of the categories and the associated content of the questionnaire.

6 Results

In total, 28 teachers answered the survey, of which 19 teach BDE. Of these 19 participants ten answered to have a degree in computer science or similar training. The most common motivations for teaching BDE were *related subject*, *professional aptitude* and/or *assignment by others* (e.g. principal).

For *related subjects*[6], the combinations were often mathematics, followed by other subjects like English or history. *Voluntary reporting* was predominantly coupled with *professional aptitude*, among those volunteers only two teachers did

[6] In Austria, the teacher training programme is linked to the choice of two (or more) subjects, which enables teaching in Austrian schools. The choice of subjects can be made independently, i.e. they do not have to be related at all.

not have proper training and came forward due to strong interest in imparting knowledge to students. Those who were assigned by others to teach BDE, did not have more than 10 years of professional experience in teaching which supports the assumption that young teachers are assigned more commonly. But this small data set cannot be used to draw conclusions about the general public. Nevertheless, the result is a good indication of an existing problem in the school setting.

More than half of the participants, which corresponds to almost three quarters of the BDE teachers, did not take part in any of our offers, but heard about our initiative through colleagues or self-referrals. In terms of previous programming experience and knowledge, one third of BDE teachers claim to have good to excellent skills. (1) Python, (2) Java and (3) C#/C++ were the most used programming languages among all our participants and are ranked by frequency. 57% of all participants state to have little to none experience at all. Slightly more than half of them are BDE teachers and the rest correspond to CS teachers. On closer inspection, these CS teachers have more than 10 years of professional experience and probably do not have a computer science degree, but attended only a university course, see figure 1.

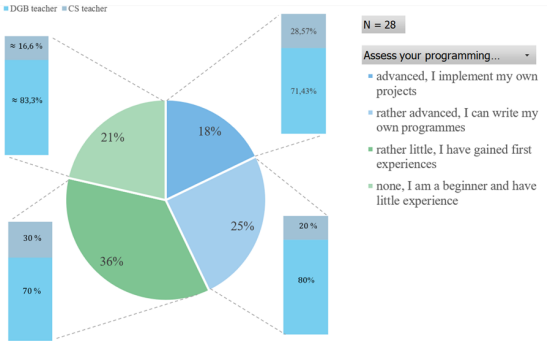

Fig. 1: Overview of programming experience of all participants.

16 out of 28 participants have little to none experience in programming, who we want to pay more attention to and reach with our programming course. For this purpose, six teachers, who already attended our course, were filtered during the questionnaire via branches. Two of these six teachers chose and completed the entire programme of our course with their students. The remaining teachers based it on the two sessions (= block 1). In the first block section the opinions on the categories *introduction to the environment* and *method selection*, e.g. rect() or triangle(), were split, with the first half describing as *suitable* and by the other as *very suitable*. Nearly 67% of the participants rated the subsection "predefined variables" as *suitable*, but no further comments for improvement were made. (1) *Problems with syntax*, (2) *Understanding error messages and finding errors*, (3) *Saving and finding files* and (4) *Lack of understanding the programme and programme process* were identified as difficult for the students

from the teachers' perspective and are ordered by occurrence. To all the concepts presented, *loops* and *compound comparisons in branches* were described as most difficult. Again only two groups have done the whole programme. A general suggestion for improvement was *interactive videos on the website* for the students to better understand the tasks and *unification and simplification of the provided cheat sheets.*

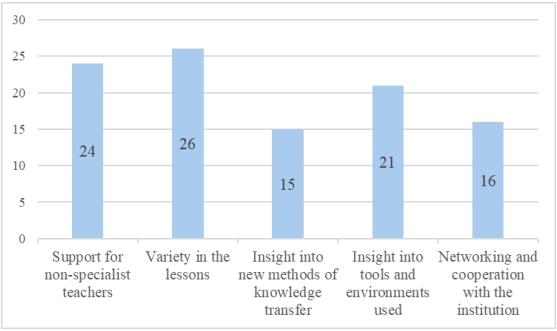

Fig. 2: Overview of programming experience of all participants.

The following items would speak for the attractiveness of the course on the part of the teachers: *Variety in the lessons*, *Support for non-specialist teachers* and *Insight into tools and environments used* (figure 2). The majority of participants would be satisfied with the existing offer and programming language, still 39% suggest a microworld, use of robots or another text-based programming language, such as Python, Rust or C#. 75% would like to receive further education or training from our institution. This can be explained by the insufficient offers, as 26.3% described the category "learning materials" as insufficient and 52.6% as neutral.

7 Discussion

This evaluation aimed to investigate the programming knowledge and needs of teachers who teach the compulsory subject BDE in secondary schools (RQ1) and to explore how we can adapt the programming course, which was originally designed from the 9^{th} grade onwards, to support these teachers and their students better (RQ2).

Regarding the first research question (RQ1), the questionnaire revealed that a significant number of teachers (57%) reported having little to no programming experience. This outcome supports our assumption that there is a potential gap in programming skills among the participants and that those untrained often face the problem in teaching programming in their classes. Furthermore, among those who claimed to have programming knowledge, the most popular

used programming languages were Python, Java and C#/C++, which is an indicator of the general popularity of the languages in today's community and underscores the choice of Processing in our course, as it is Java-like. Regarding RQ2, the questionnaire results shed light on the challenges faced by teachers and their suggestions for improvement in the new version of the programming course.

Difficulties for students included *syntax problems*, *understanding error messages*, *saving and finding files*, and a *lack of understanding of the program and its process*. This results from oral comments made by students and their teachers. Teachers observed that loops and compound comparisons in branches were particularly challenging for the students. Feedback from teachers who had already participated in the course highlighted the need for *interactive videos* on the course website and *simplified cheat sheets*. These insights can guide future improvements to the programming course, addressing the identified challenges and suggested improvements. There were also suggestions for additional enhancements, such as including microworld, using robots, or incorporating alternative text-based programming languages like Python, Rust or C#. These suggestions indicate a desire to further diversify and customise the course content to teachers' and students' specific needs.

The questionnaire also revealed positive aspects of the course that were attractive to teachers, such as the *variety of lessons* or *support for non-specialist teachers*. A very satisfactory result on the course, since it was the goal to reach the teachers that needed it and give students a good first programming experience. However, it is crucial to note that the study's limitations, particularly the small sample size of only 28 teachers, strongly restrict the generalisability of the findings. A larger sample would provide a more representative picture and offer opportunities for future research.

8 Conclusion

In conclusion, this work gave an overview of the structure and content of the newly introduced subject "Basic Digital Education" (BDE) in secondary schools and the associated difficulties faced in teaching, especially in the area of programming. In addition, this paper presented an already existing programming course from 9^{th} grade onwards, which shows great potential to be applied in lower levels. The survey results highlighted a programming skill gap among teachers responsible for teaching BDE classes and identified certain concepts as difficult for students.

The feedback from teachers who had already attended the programming course provided valuable suggestions for improvement. However, the study's limitations, particularly the small sample size, call for further research to validate and extend these findings. Future investigations should aim to include a more extensive and more diverse sample of teachers to obtain a more comprehensive understanding of the programming knowledge and needs of teachers in secondary schools. In the future, we would like to adapt the suggestions for improvement

in the current course and adapt it for the 8th grade, hold further training for CS and BDE teachers with a focus on "programming" and offer programming courses from 5th grade.

References

1. Brinda, T., Brüggen, N., Diethelm, I., Knaus, T., Kommer, S., Kopf, C., Missomelius, P., Leschke, R., Tilemann, F., Weich, A.: Frankfurt-Dreieck zur Bildung in der digital vernetzten Welt. Ein interdisziplinäres Modell (2020)
2. Bundesministerium für Bildung, Wissenschaft und Forschung: Änderung der Verordnung über die Lehrpläne der Mittelschulen sowie der Verordnung über die Lehrpläne der allgemeinbildenden höheren Schulen . https://www.ris.bka.gv.at/eli/bgbl/II/2022/267/20220706 (2022), [Online; accessed 25-May-2023]
3. Bundesministerium für Bildung, Wissenschaft und Forschung: Mini-mooc ,,digitale grundbildung". https://www.bmbwf.gv.at/Themen/schule/zrp/dibi/paed/mooc_dgb.html (2022)
4. Lakanen, A.J.: On the impact of computer science outreach events on k-12 students. Jyväskylä studies in computing (236) (2016), https://jyx.jyu.fi/handle/123456789/49729
5. Landman, M., Futschek, G., Unkovic, S., Voboril, F.: Initial learning of textual programming at school: Evolution of outreach activities. Olympiads in Informatics pp. 43–53 (2022). https://doi.org/10.15388/ioi.2022.05
6. Moritz, T.: Bildung und medienpädagogik im zeitalter der digitalen medien. Probleme, Herausforderungen und Perspektiven für Pädagogik, Bildung und Schule in Zeiten von Internet und Telekommunikation. Medien Impulse pp. 51–60 (2001)
7. OECD: Curriculum overload: A way forward. OECD Publishing, Paris (2020). https://doi.org/10.1787/3081ceca-en
8. Papert, S.: Microworlds: Incubators for knowledge. Mindstorms-Children, Computers and Powerful Ideas pp. 120–134 (1980)
9. Porter, L., Guzdial, M., McDowell, C., Simon, B.: Success in introductory programming. Communications of the ACM **56**(8), 34–36 (2013). https://doi.org/10.1145/2492007.2492020
10. Reas, C., Fry, B.: Processing: A programming handbook for visual designers and artists. MIT Press, Cambridge, Massachusetts (2007), https://ieeexplore.ieee.org/book/7008153
11. Sentance, S., Csizmadia, A.: Computing in the curriculum: Challenges and strategies from a teacher's perspective. Education and Information Technologies **22**(2), 469–495 (2017). https://doi.org/10.1007/s10639-016-9482-0, https://link.springer.com/article/10.1007/s10639-016-9482-0#citeas
12. Vegas, E., Hansen, M., Fowler, B.: Building skills for life: How to expand and improve computer science education around the world. Brookings (2021)
13. Wu, L., Looi, C.K., Multisilta, J., How, M.L., Choi, H., Hsu, T.C., Tuomi, P.: Teacher's perceptions and readiness to teach coding skills: A comparative study between finland, mainland china, singapore, taiwan, and south korea. The Asia-Pacific Education Researcher **29**(1), 21–34 (2020). https://doi.org/10.1007/s40299-019-00485-x, https://link.springer.com/article/10.1007/s40299-019-00485-x
14. Zhang, L., Nouri, J., Rolandsson, L.: Progression of computational thinking skills in swedish compulsory schools with block-based programming. In: Proceedings of the Twenty-Second Australasian Computing Education Conference. pp. 66–75 (2020)

A Appendix: Excerpt from the questionnaire

1. Indicate how long you have been teaching:
   (a) 0 - 1 years
   (b) 1 year
   (c) 2 years
   (d) 3 years
   (e) 4 years
   (f) 5 years
   (g) 5 - 10 years
   (h) 10 - 20 years
   (i) 20 - 30 years
2. Please state which subjects you teach:
   (a) geometry
   (b) mathematics
   (c) English
   (d) German
   (e) other living foreign language
   (f) Latin/ancient Greek
   (g) Computer Science
   (h) history
   (i) geography
   (j) Basic Digital Education
   (k) other:
3. Why did you decide to teach "Digital Basic Education"? (*Multiple selection possible*)
   (a) Voluntary reporting
   (b) Professional aptitude
   (c) Related subject
   (d) Classification by other persons (e.g. principal, etc.)
   (e) other:
4. Assess your programming skills:
   (a) none, I am a beginner and have little experience
   (b) rather little, I have gained first experiences
   (c) rather advanced, I can write my own programs
   (d) advanced, I implement my own projects
5. If applicable: Which programming languages have you already worked with? (*Multiple selection possible*)
   (a) Java
   (b) C#/C++
   (c) Python
   (d) Processing
   (e) PHP
   (f) other:
6. Please indicate why you would/have chosen the programming crash course (an introduction to programming with a Java-like programming language and comprises 1-2 double lessons):(*Multiple selection possible*)

(a) Students should get to know a new programming language
(b) Processing as an introductory language to programming for students
(c) I have little programming experience myself and would like to see how it is implemented.
(d) Future cooperation with the institution
(e) Part of the syllabus
(f) other:

7. Please indicate if you have already booked a Processing course:
(a) 2 hour session
(b) 4 hour session
(c) No
8. I found the content in the first 2 hour session suitably prepared:
(a) Introduction in the environment
(b) Method selection (e.g. rect(), triangle())
(c) Pre-defined variables
9. I found the content in the second 2-hour block suitably prepared:
(a) Branches - simple comparisons
(b) Loops
(c) Branches - compound comparisons
10. Indicate whether difficulties were noticeable in your class: (*Multiple selection possible*)
(a) Dealing with the environment
(b) Problems with the syntax
(c) Understanding and finding error messages
(d) Difficulties in creating programs to solve the tasks
(e) Saving and finding files
(f) Overstretched by the resources
(g) Lack of understanding the program structure
(h) other:
11. Please mark which concepts have been noticeably not easy for the students: (*Multiple selection possible*)
(a) Branches - simple comparisons
(b) Loops
(c) Branches - compound comparisons
(d) Variables
(e) Methods
(f) other:
12. Rate this programming crash course. This offer would be good because (*Multiple selection possible*)
(a) Support for non-specialist teachers
(b) Variety in the classroom
(c) Insight into new methods of knowledge transfer
(d) Insight into tools and environments in use
(e) Networking and cooperation with the institution
(f) other: