DUT Dansk Universitetspædagogisk Tidsskrift
Læringsmiljøer: Relationer & Rammer
Årgang 19, nr. 36, 2024

Faglig artikel, fagfællebedømt

Creating a motivating environment in a programming course using a two-track exercise split

Sine Zambach[1], Department of digitalization, Copenhagen Business School

Abstract

In our society, programming and IT literacy are important skills. However, in introductory courses in higher education, there is a struggle to design the coursework in a way that engages and motivates both beginners and people who already know some code.

This paper presents a case study that explores the design of optional exercise sessions as a shorter 'fast track' for experienced or well-prepared students, and a longer 'normal track' for those who need help with the technical parts of the coursework. The case is analysed using self-determination theory to investigate student motivation in such a design.

Students perform well and are generally happy to be able to choose between tracks to find their own fit. Whether a 2-day induction workshop or a track split leads to the best learning environment is a question for future research.

Problem

In most higher education courses, there is a diversity of learners with different backgrounds and learning preferences. Therefore, these courses should strive to motivate this diversity of learners. Programming courses in particular, which have evolved rapidly in recent decades, fail to do this, and they also suffer from a lack of female participation (Jenkins et al., 2002; Robins et al., 2003; DEA & Microsoft, 2019).

This is particularly problematic, since coding and computational thinking are increasingly valued skills in an increasingly digitised world, and since large groups of talent will be lost if their diversity of skills is not catered for in the classroom. To avoid losing the great potential in skills development, even more critical in today's increasingly digital society, we must investigate methods for developing the skills of students and share reflections on these investigations. It is vital not to discard valuable opportunities in the process.

This paper will focus on how to create an environment that embraces different levels of programming skills, thereby creating a more motivating educational environment.

[1] sz.digi@cbs.dk

Literature

There are different approaches with respect to handling the large differences in programming skills found in introductory programming and data science courses in higher education (Alvarado, 2018; Jenkins & Davy, 2002; Grabarczyk et al. 2022; Cohoon & Tychonievich, 2011). In particular, I will highlight two different proposed solutions to which I will return in the Discussion section.

One solution is to provide a special onboarding session of a few days' duration prior to a full education programme (Grabarczyk et al. 2022). Grabarczyk et al. have designed a three-day onboarding course prior to a full bachelor programme at the IT-University of Copenhagen, during which the students will learn basic programming before starting the bachelor programme. In their study, the students participating in the onboarding course were able to catch up to the level of their peers, and the dropout rate improved. In addition, the students highlighted higher confidence and self-efficacy.

Another option is to develop a specially designed 'low speed' course that runs alongside a regular course, with the aim of recruiting students into computer science majors (Cohoon & Tychonievich, 2011). One example of this is the special low-speed course developed by Cohoon and Tychonievich which had simpler tasks and a more thorough introduction to the programming concepts.

These earlier approaches have had positive effects on gender and minority equality and on students without prior computer science experience – both are issues often associated with education programmes in computer science. However, neither of these approaches is flexible during the course. If, after two lectures, a student realises that they would have benefitted from the introduction or the low-speed course, they cannot change their decision.

Initiative

The work presented in this paper is based on an introductory programming course at a Danish university and proposes a third solution: to have two different exercise tracks within one course. The course is a programming course for absolute beginners, taught in English, and the two tracks were designed so that both true beginners, students who already knew some programming, and fast learners would get the most out of the teaching. The course is described in the case study description below.

The two-track course design is part of an experiment performed in an introduction to programming course in the fall of 2021 and 2022 to investigate how to increase the student motivation through autonomy, competence matching and relatedness among the students, as well as how to enhance overall performance of students' performance.

This led to the development of the following research questions:

How does the introduction of a fast and a normal track support student motivation and satisfaction with the content and learning in introduction to programming courses?

Can you enhance student performance in introduction to programming courses by differentiating the exercise frames using a fast and a normal track?

In this article, I will discuss the two-track approach in relation to the earlier experiments, particularly the approaches by Cohoon & Tychonievich (2011) and Grabarczyk et al. (2022), mentioned above.

The case will be discussed from the perspective of self-determination theory according to which autonomy, a feeling of competence, and relatedness are core psychological needs for intrinsic motivation for learning and well-being (Niemiec and Ryan, 2009; Mishkin, 2019). These core needs are based on Niemiec and Ryan's (2009) work: You have autonomy when your actions are in accordance with personal values and interests, and when you are feeling a sense of ownership and control over your behavior and decisions, which can lead to higher intrinsic motivation. The feeling of competence is the need to master tasks, learn new skills, and overcome challenges. The concept is linked to efficacy and accomplishment, since it fuels motivation and confidence. Third, relatedness is the sense of belonging in social interactions in class. The quality of the social interactions has an impact on well-being and motivation.

In addition, I will present some of the practical challenges and advantages that the two-track solutions cause, and I will argue why this two-track approach is sometimes to be preferred. This is particularly relevant when we focus on the notion of students as agents who want to have an autonomous influence on how and when they learn (Bandura, 2006).

This case study can serve as inspiration for teachers designing or redesigning introductory technical courses in which students often have very different levels of skills at the beginning of the course.

Context

The course analysed in the case study, Introduction to Programming and Data Analysis, is an elective aimed at bachelor-level business students with no prior experience in programming or data analytics. The course has been running for two semesters, has had between 86 and 105 participating students, and is designed and refined by the author in collaboration with various instructors. The course is part of an 'IT-minor' required for accessing the master programme in Business IT at a higher education institution.

The course was designed in 2021 as a course that would introduce absolute beginners from several bachelor programmes at the institution to both programming and data analytics. Further, it has a business aspect in that the students must reflect on the usage of the tools they learn within a business case of their own choice. The target groups included students taking the course as part of their minor to study Data Science or Business-IT afterwards, students intending to study one of the typically technically demanding Finance master programmes, as well as students who simply wanted to explore their analytical skills. Data science is one of the top skills wanted in business, measured in employment and salary. Since the course was an elective, students from multiple study directions participated, including exchange students (2% in 2021 and 37% in 2022) and Danish students. Women made up approximately 45% of the student group.

The course contains 20 hours of face-to-face lectures (10 sets of 2 lectures on different topics, including a few small tasks and mainly PowerPoint presentations), 10 hours of online instruction (small technical videos, typically 6–15 minutes each), and 10 sets of 3 hours of in-person exercises during which students complete programming exercises that are also discussed by the instructor. For the fast track, exercises are 10 sets of 2 hours, approximately.

For the exam project of the course, the students were required to find their own data to analyse, so that they could write their final exam within their own areas, whether that was finance, international politics, or creative business. I emphasised that the students should help each other in their programming, since you learn a lot both from getting help from your peers and by helping your peers.

However, although I tried to accommodate the diverse range of student types, I noticed one significant problem in class. There was a huge difference in the background skills of the students at the start of the course (figure 1 A and B). Some already had programming experience while others had none.

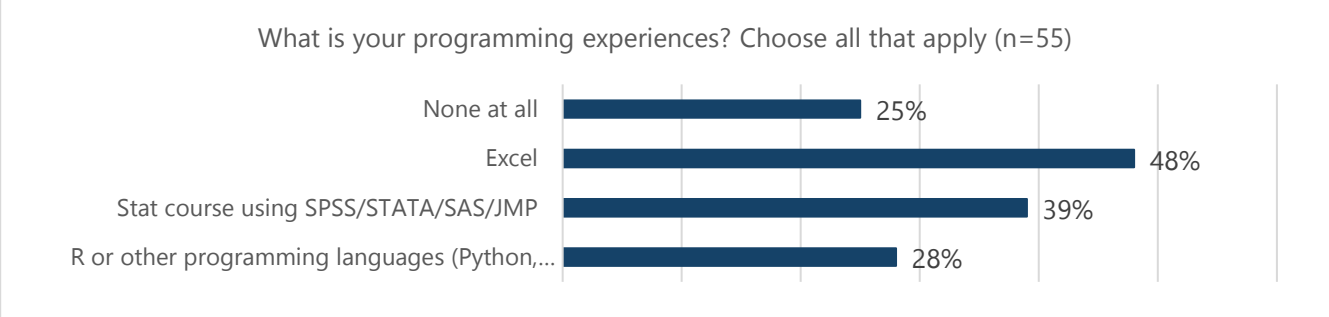

Figure 1 A: Students' experiences when entering the class in 2021

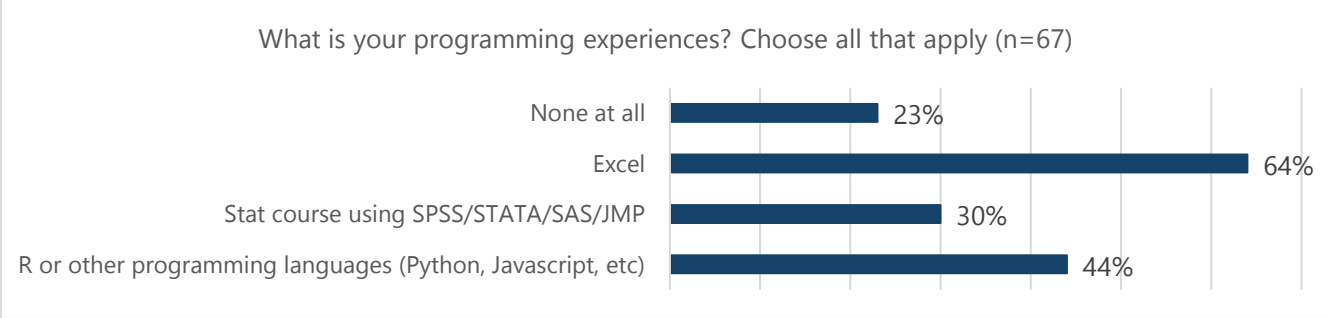

Figure 1 B: Students' experiences when entering the class in 2022

This resulted in two exercise classes where both instructors reported frustration from two sides of the classroom: frustration among students who already knew how to program when we used extra time on the exercises; and a high level of frustration among students who were new to the field and had no prerequisites when we increased the tempo. This is described very well by Jenkins & Davy (2002) who name the two groups 'the strugglers' and 'the rocket scientists'. Instead of feeling related, they felt disconnected, and instead of feeling competent, the strugglers felt incompetent, while the rocket scientists felt a lack of autonomy; these are all factors that affect intrinsic motivation and self-determination. Therefore, I introduced an experiment which will be described in the implementation section below.

Implementation

A case study design was used to investigate an intervention in the course described above, specifically the introduction of two different exercise tracks. The success of this intervention will be evaluated using grade

measures, comments from students' evaluation of the teaching, and three in-depth interviews regarding the students' competencies, feelings of relatedness, and autonomy.

Intervention: Introducing a fast track and a normal track

After the first three weeks of teaching the 2021 course, I decided to change the exercise settings so that we had two different types of exercise classes. One was called the fast track and the other was called the normal track (originally it was called the slow track). Since the course is designed for beginners, it is important to have a positive term for the normal track, so that the name does not imply that it is for poor students. The decision was based on the challenges described above, and students' different levels of experience illustrated in Figure 1 A. The figure is based on data from a questionnaire that students were asked to fill in before the first lecture. Both tracks introduced the same exercises; the main difference was the classroom format.

The fast track was presented as a class for students who had already completed the exercises and simply wanted to hear the solutions presented. Classes typically lasted between one and two lectures (1-1.5 hours). It was thus suitable for individuals who preferred to work with the exercises beforehand, as well as for individuals who were confident enough in their programming ability to complete the exercises while they were presented during class.

The fast track provided a sense of autonomy. Individuals who prefer a fast track may be attracted by the autonomy to set their own pace and challenge themselves according to their preferences. They are also given the opportunity to improve quickly, which may be in line with their need for competence. Finally, individuals who prefer a fast track may feel that they belong more to the class since they are working at the same pace as their peers.

The normal track was presented as a class in which extra time would be spent on presentation, solving each exercise, and discussing the solutions. It would cover three lectures (2.5 hours). For me as a teacher, it was particularly important that this space was inclusive and allowed for 'stupid' questions. It was also important that almost no one got frustrated by the 'slower' speed, since they could just switch to the fast track. This track attracted absolute beginners as well as some individuals who liked the opportunity to solve the problems in class and share their results with others for peer feedback.

Students who chose the normal track would find a sense of competence in gradually building their skills and a sense of connection with others who were also struggling to learn how to program.

Students were able to choose the track on a week-by-week basis, so that they could take advantage of the most appropriate offer for the given period, thus satisfying their need for autonomy.

Implementation of the tracks

In the first year (2021) I took advantage of the fact that the fast track instructors could join the normal track when they had finished the fast track class. In the second year (2022), I also had a student instructor in the normal track to assist students who had technical issues, so they could receive support more quickly.

Communication

In the second year, it was pointed out to me that the names I had initially devised (slow track and fast track) were a bit of a nuisance. Slow was intended to connote 'slow-moving' or 'reduced speed' but might also imply 'dumb'

or 'slow to understand', especially in American English, so I renamed it 'normal track'.

It is crucial to communicate to students that the normal track is the standard, while the fast track is meant for students who already possess some coding knowledge or have completed the exercises before the class. In this way, the inexperienced students for whom the course was originally designed are not alienated, and students are given the agency to choose for themselves.

The more experienced students usually take the course because of the machine learning topic that is introduced at the end of the course, which is often considered advanced. In addition, we had extra exercises for them to do during the course, which were both challenging and application-oriented, and therefore hopefully related to their main areas of interest and their need to develop their competencies.

Interviews

From the anonymous evaluation comments I could see that the students only had positive comments about the fast track and normal track, and the course received an average of 4.3 out of 5, which is generally considered good for a programming course. The four comments in 2021 and six comments in 2022 regarding the tracks were very brief and general, for example, "I liked how there was a slow track and fast track in the course", and thus too sparse for a qualitative analysis. Some of this was discussed in classes, but still at a general level, and the answer rate for the course was 24% and 41%, respectively. Therefore, I have supplemented the evaluation with semi-structured, in-depth interviews with three students.

The three students chosen for the interviews were all enrolled in a bachelor programme on their 3rd semester, and table 1 shows information about them. The author conducted the interviews in English and Danish, and the interview guide is listed in table 2. The data were transcribed and coded using the three core concepts mentioned above: autonomy, competencies, and relatedness.

Student	Sex	Age	Nationality	Year	Programming Experience	Track
S1	Male	25	Danish	2022	Yes	Fast track
S2	Female	24	Non-Danish	2021	No	Normal track
S3	Male	24	Danish	2022	No	Changed between fast track and normal track

Table 1: The demographics, programming skills, and track choice of the interviewees

Question	Purpose
Which competencies did you have before and after the course	To identify if they had diverse learning outcomes and what their skill level was before the course started.

What were your expectations for the course?	To ensure alignment with the course description.
Which track(s) did you follow? And how did you find them?	To ensure we have a representative from each track, and to examine experiences with autonomy in this course.
What were your experiences with the two tracks (both academically and socially)?	To have a broader discussion about the tracks and their impact on learning. As well as to explore feelings of relatedness.
Were there other ways in which differentiation could have been achieved? Full courses on different levels? Two-day brush up?	To hear opinions and other experiences with differentiation.
Do you have any other comments?	To offer the option of raising new topics or discussion points concerning the course or similar.

Table 2: Questionnaire for semi-structured in-depth interviews with 3 business school students

Results

The analysis of the semi-structured interviews found evidence of the following aspects from self-determination theory (Niemiec and Ryan, 2009):

- Autonomy – the option of choosing between tracks on the course.
- Competencies – the actual capabilities after the course
- Relatedness – the social values of the course.

Below, I will unfold these three themes whilst also incorporating evidence from student evaluations of the teaching, and performance indicators.

Autonomy

According to the students, the two-track design was valued highly due to the perception of choice they had each week, which they found highly motivating. This is reflected in the following quotes, which provide insights into students' experiences with autonomy. S1 states: "If there was no fast track, you would have ended up losing those who already knew something". And S3, who changed between the two tracks, was also happy about this option "Fast track was quite awesome, but you had to be well prepared. The normal track was sometimes a bit too slow for my temperament". Meanwhile, S2 appreciated being able to choose the slower option: "As it was my first course in (the programming labguage) R, I would rather go slowly. Maybe I would have chosen the fast track if it was my second R course." Autonomy in this case is reflected in the students' appreciation of the freedom to choose a learning pace that suited their individual needs and levels of prior knowledge, which were perceived dynamically by the students during the course.

Additionally, we discussed other options for differentiated teaching, and S2 and S3 (beginners) both expressed that a short brush-up workshop for beginners would have been nice. S2 even suggested that there should be both a beginner's workshop and the normal track option throughout the course. This expresses a preference for having

several choices in their learning experiences, suggesting that autonomy involves not only the pace of learning but also the type of support or additional resources provided.

The positive impact of incorporating autonomy into the learning environment was also underscored in the student evaluations from the teaching surveys from both 2021 and 2022: there was general satisfaction with the format, and the students expressed satisfaction with the fact that the exercises were divided into two different types of tracks. One example of such a comment is: "I liked how there was a slow track and fast track in the course." This also reinforces the importance of autonomy in learning since it aligns with the theoretical self-determination principle that autonomy supports motivation and well-being.

In summary, the concept of autonomy is reflected in the students' positive response to having choices in the form of various tracks and additional learning options. Autonomy, in this context, contributes to the students' motivation, satisfaction, and sense of control over their learning experiences.

Competencies

The value of the exercises with respect to the development and manifestation of the students' competencies was also reflected in their choice of track. While S2 appreciated the gradual learning process in the normal track, and that the instructor talked about the solutions and code snippets in plenum, S1 mentions that he enjoyed the fact that they had to articulate their solutions and discuss them on the fast track. Likewise, competencies before the course allowed students to develop competencies in diverse ways, from understanding R, how it is structured, and how you program, mentioned by the two beginners, to S3 who described learning more advanced skills: "To communicate and create a story from the actual analysis we were doing in the exam project."

The ability to communicate and create narratives from analysis indicates a progression in competencies beyond technical knowledge to more advanced skills involving communication and storytelling based on data analysis.

The distribution of grades might provide a quantitative proxy for the effect of the two-track approach on the students' performance, since students write a report and are graded on an equal footing. As teachers and instructors, our impression of the students (also informed by the data in Figure 1) was that they mostly had a low or a high level of skills at the beginning of the course. However, rather than a lot of low grades and many high grades, the distribution of the grades forms a skewed "bell curve" (Figure 2) which may indicate that the beginner students have, to some degree, fulfilled their potential. An improvement can be seen in 2022, which may reflect that the course was taught for the second time, and therefore most teething problems were fixed by 2022.

Another possible measure of the effect of the course is student evaluations of the teaching. The question "The course has increased my knowledge of the subject" received a score of 4.5 out of 5 in 2022, even though many students had programming experience prior to starting the class. Other students with less experience mentioned good aspects of the course: "I appreciate that the course was designed for people with no previous knowledge of coding. Learning an actual new skill was great.", which expressed self-efficacy and motivation. This feedback underscores the course's positive impact on the students' motivation and perceived self-efficacy. Comments appreciating the course design for individuals with no previous coding knowledge highlight the motivation generated by learning new skills. Likewise, the knowledge increase despite prior programming experience underscores the course's positive impact on students' motivation and perceived self-efficacy.

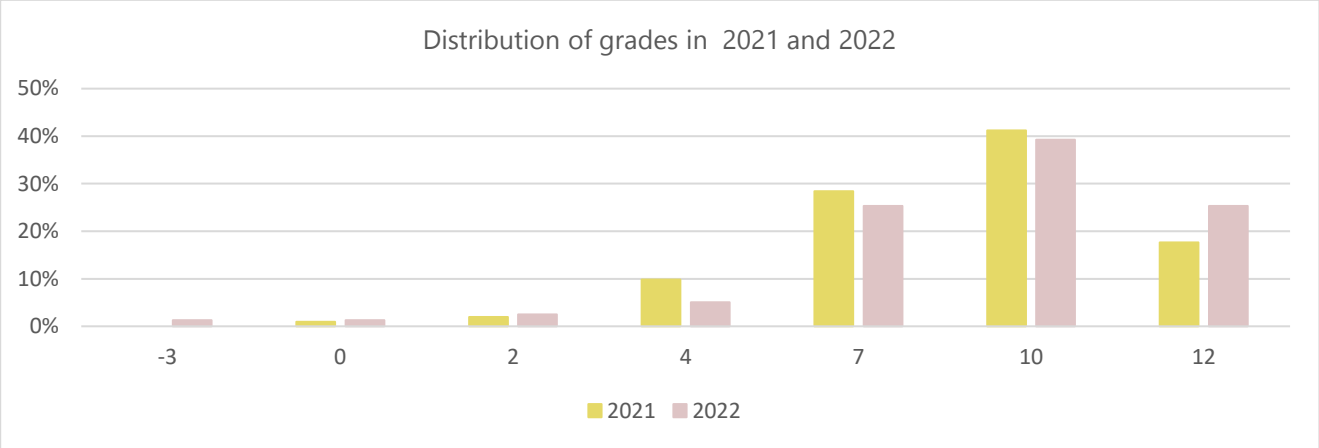

Figure 2: Distribution of grades at the final exam in the fall 2021 and 2022. Both ordinary and retake exam grades are included in the total of 102 in 2021 and 79 in 2022. 4 did not submit a final report in 2021, 7 in 2022

The negative comments regarding competencies were mainly about the difficulties with finding a case/getting more supervision for their final project (5 comments), and that the pace was either too slow (2 comments) or too fast towards the end (3 comments). This suggests the importance of aligning course elements with the students' needs to better support competency development.

Relatedness

On the question of relatedness and social connection, I mainly focused on evidence from the interviews. The students who claimed they had the least interaction (S1) mentioned that there was a Facebook thread, but in the fast track class, people mostly sat with their own computers. S2 and S3 got much more out of the social situation in the classroom and used their peers to discuss solutions, brainstorm, and discuss the lectures. However, this was not facilitated very much in class, and as S3 suggests, "maybe a bit more group work and mini cases would have brought us together". In the student evaluation of the teaching survey, one student added that "the dynamic of classes could be improved". Among the components of self-determination theory, relatedness was the one least affected by the course. Despite some degree of social interaction, the overall impact of the course in terms of fostering a sense of relatedness among students seemed limited.

Discussion

The discussion will focus on the main issues, diversity in experience, evaluation, and performance, and what may work best as a teaching practice in different scenarios.

Comparing the case with other initiatives of differentiated teaching

The normal way of handling diversity in experience seems to be to have a brush-up course or an onboarding course (Grabarczyk et al., 2022; Cohoon & Tychonievich, 2011). This can be very convenient, since the typical introduction to programming course often assumes that all students are more or less at the same level. In addition,

for a full study programme, as in Grabarczyk et al. (2022), it makes sense to ensure that everyone starts on the same page to support student confidence and self-efficacy throughout the rest of the programme.

Compared to a brush-up course at the start (Grabarczyk et al., 2022), spreading out the upskilling of the students with less experience across all exercises offers greater flexibility and autonomy for both students and instructors. This means that it is not necessary to carve out a week from one's schedule, and the programming concepts can slowly be adopted and understood by the students. This supports the students' self-efficacy as human agents (Bandura, 2006) who can shape their learning themselves and grow during the semester. This may not apply for an intensive course which must be chosen even before the lectures have started. In terms of competencies, this option also introduces the student to many new concepts which can be overwhelming in such a short period of time.

Further, even low-skilled students may differ in their ability to adopt the content. Some may want to follow the fast track after a few classes, while others may find themselves on too fast a track after a few weeks and want to change from fast track to normal track. For these students, the two-track design of my course offers full flexibility and thus autonomy, and a match with their individual development of programming competencies.

A fixed division of learners, as suggested by Jenkins & Davy (2002) and Cohoon & Tychonievich (2011), on the other hand, may be too rigid, since not everyone will remain 'strugglers' or 'rocket scientists'. Interestingly, some of the 'rocket scientists' also seemed to prefer the normal track exercise classes, and some beginners preferred fast track exercise classes. With the two-track design, students have the option to autonomously make a choice every week and are not reliant on what was perhaps a bad choice that they made at the beginning of the course or even before.

Cohoon & Tychonievich (2011) experimented with this, developing a full parallel course to the regular Computer Science 1 (CS1) called CS1X, which assumed fewer programming skills from the students. This required a complete course designed for beginners only, in parallel with a course designed for students who already had some programming experience. This might lead to a higher degree of relatedness among the students, since they have more similar needs and can see themselves in both the lectures and the exercises.

The two-track model allows for changes during the course and is therefore more flexible and also more economically feasible than two parallel courses – particularly if the allocation of rooms and teachers is optimised as in figure 2 B above. And the students will experience similar competencies and relatedness in exercise classes as well, though not in lectures, where they are mixed.

The students I interviewed said that they would like even more options to choose from, including a fast track, a normal track, and a 2-day introductory course. However, as a university, we also need to remain sustainable in terms of the hours we allocate to teaching, since few universities can afford to give special treatment to programming classes. It would be interesting to investigate further the impact of different initiatives and what is most efficient in terms of both motivation and performance.

Finally, we must assume that the great divide in programming experiences will continue as long as there are almost no formal programming classes in Danish schools or high schools. Thus, the competency level of students will not be the same, and motivation will be an ongoing problem in this regard. Relatedness will also be difficult for

minorities, as suggested by Mishkin (2019). Therefore, we must continuously ensure that as many motivational factors as possible are present in the classroom when we teach courses containing an introduction to programming at an introductory level. This may be in the form of more group activities in class, optional as well as mandatory, more feedback and space for diverse perspectives.

Evaluation and performance

We asked the students to assess whether they liked the split into two tracks. To summarise, based on the student evaluation of the teaching from both 2021 and 2022, there was a general satisfaction with the format and with the fact that we split the exercises into two different types of tracks. The same holds true for the three interviewees. It would have been interesting to ask a larger number of students whether they would prefer, for example, a brush-up course at the beginning of the course. This question may be included as an extension of the systematic course evaluation questionnaire.

In general, it is interesting to investigate what the best model would be for teaching introductory programming in higher education under different circumstances. What do the students like the most? What do the teachers prefer? What results in the best student performance? And how do we keep courses sustainable with respect to university economy?

Unfortunately, it has not been possible to study the effect of the tracks on individual exam grades, since the choice of track is not linked to the student ID. Moreover, students were offered a flexible approach, and not everyone has consistently selected one of the tracks. However, we do have indications in this study, based on completion rate and grade distribution, that the students perform better when diversity is considered in the course design.

A proxy for the impact of the tracks on motivation, competence, relatedness, and autonomy might be measured specifically by asking about the interpretation of the tracks in the systematic course evaluation in the future.

Designing a randomized controlled study to measure a causal effect will require a parallel course in which there is no choice of joining different tracks, or, perhaps more ethically, an A/B-test in line with for instance Tomkin & Charlevoix's (2014) approach with a 2-day onboarding workshop versus a fast/normal track. This is more ethical, since both options in principle support self-determination, but we do not know which one is the best solution. Students should then be randomly assigned to one of the two courses, and the well-being and performance of the students should be measured for those on the normal track versus the fast track and compared to the well-being and performance of the students in the course section where everyone gets the option of joining an onboarding class. This may be carried out in future work.

Administrative setup

For the course in this case study, department administration was responsible for the puzzle of booking rooms and scheduling lectures for the teachers. It is important that they understand the system, so that teachers, rooms, etc., are all aligned, since the rooms and the teachers' schedules may be constrained in such a way that the suggested solutions may not be possible. In addition, the design may influence students' teacher evaluations: Students should be allowed to evaluate teachers from all exercise classes since we do not know which track they followed, or if they followed them both interchangeably. Considerations about student evaluations were continuously discussed with

the administration, particularly the first time we taught the course.

To sum up, there are some practical considerations to take into account in the implementation of a two-track exercise design.

Rooms. I recommend having rooms next to each other and conducting the sessions at approximately the same time. I had two rooms in parallel, which meant that the instructor from the fast track could come in and help the normal track, and the students did not have to choose tracks until the last minute, as illustrated in the schedule in figure 3 A.

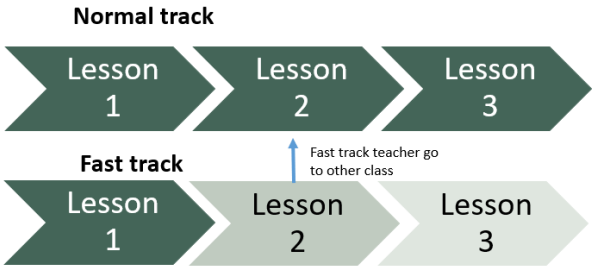

Figure 3 A: Schedule for the two exercise classes in 2021 and 2022

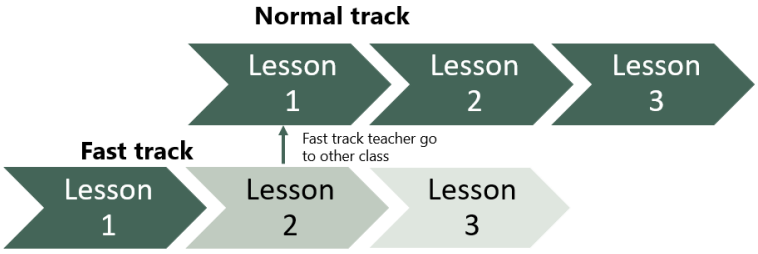

Figure 3 B: Suggestion for a staggered, alternative time schedule for two exercise classes

Teachers. In 2022, the course had 86 students enrolled (out of a maximum of 120) and this allowed for two exercise classes and one extra instructor. In total, we were three instructors: two for the normal track and one for the fast track. In the normal track, the main lecturer gave instructions, supported by a student instructor, while in the fast track, an external lecturer (with industry experience) gave instructions.

If fewer students had enrolled, e.g. around 60, the course might have had a setup with only two exercise teachers, using the model in figure 3 B. If the maximum of 120 students had enrolled, we might also have needed the schedule in figure 3 B, with three instructors during the first and busiest lecture.

Gender considerations

Typically, in introductory programming classes, there will be very few female students, and they often have the least amount of prior programming experience because they have not – to the same extent as male students – coded for fun for video games, etc. (Fisher & Margolis, 2003; Borsotti, 2018). A small number of any minority in a

class will easily lead to even fewer participants of that population due to the demotivating lack of relatedness (Mishkin, 2019). This course had approximately 45 % female students in both 2021 and 2022, and therefore we discuss the measures introduced for refinement and inspiration in relation to supporting diversity in programming work below.

We see gender diversity both among students as a whole (~40-45 % women) and among the less experienced students (of which 9/16 were women in 2022) as well as among the experienced students, who have prior experience with programming (of whom 15/31 were women in 2022) (from figure 1). Students were not asked if they thought they were good at programming themselves, which might have yielded different results, since women typically underrate themselves (Bundsgaard et al., 2019; Eickelmann et al., 2019). Therefore, I purposely asked students in class about their specific experiences, which seemed to create more gender-neutral answers and indicated an equal level of competence among genders, despite a high variance.

The high percentage of female students on the course in general (45 % is considered high within computer science classes) can also be explained by a preselection, in that people already consider the elective a CSX-like course (Cohoon & Tychonievich, 2011) since the course description was written carefully with inclusive language. In addition, other issues such as topic selection have an effect (Marcher et al., 2021): I included more business-related topics and fewer topics focusing on unrelated or abstract examples, and the gender of the main teacher (female) may also have had an effect.

An online version

How can we use the fast track and normal track thinking in a blended or online setting? One obvious idea might be to simply record a presentation of the exercises after each class. The normal track could then take place synchronously, either as a face-to-face activity as part of a flipped classroom, or online in a Zoom meeting with breakout rooms where students could work in groups of 2-4 in each room. This would make it easy for students to ask for help and for instructors to help those who need it. Students in the fast track could just watch the video to check if they are on track with their solutions. This solution can be relevant for pandemics, an urgent need for upgrades of programming competencies for lifelong learning initiatives, etc.

Limitations

Finally, I would like to address a few limitations of this study. Besides the obvious measures that should be included in a future study as mentioned above, I have only interviewed three students. Although the interviews are relatively representative with regard to track selection, study and gender, and are also supported by written student evaluations, they cannot capture the full range of experiences of the course. As such, the interviews can mainly provide a few qualitative insights, and a larger sample will be needed in future research. Further, other factors in addition to the form of teaching may be important, such as instructors, classroom constellations, or COVID-19.

Conclusion

In this study, I have presented a two-track design of an introductory programming course, taught in the fall of 2021 and 2022. The two-track design led to a high degree of autonomy and flexibility for the students, who often have different levels of programming skills at the beginning of such a course. The tracks also introduced a degree

of relatedness, particularly in the exercise tracks. Students could choose tracks from week to week according to their competencies (i.e. both experience and level of preparation). This suggests a higher motivation and satisfaction with the content and learning, answering research question 1, *How does the introduction of a fast and a normal track support student motivation and satisfaction with the content and learning in introduction to programming courses?*

Further, in relation to research question 2, *Can you increase student performance in introduction to programming courses by differentiating the exercise frames using a fast- and a normal track?* the study indicates that in general, the students performed better and had a higher completion rate than what is normal for introduction to programming classes. Moreover, the solution was economically feasible for the university.

The design is as follows:

- Include two different exercise classes for the course, one of shorter duration and one of longer duration.
- The exercise class of longer duration should allow sufficient time for solving exercises as well as for thorough discussions of solutions and applications.
- The lecturer should ensure communication of the purpose and flexibility to both students and co-instructors.
- Collaborate with administration to design a time schedule for the course that is as appropriate as possible within institutional constraints.

The course appealed to a relatively gender-balanced population of students and seems to lead to better intrinsic motivation and feelings of well-being in the two pilot studies included.

References

Alvarado, C., Umbelino, G., & Minnes, M. (2018, February). The persistent effect of pre-college computing experience on college CS course grades. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 876-881).

Bandura, A. (2006). Toward a psychology of human agency. *Perspectives on psychological science*, *1*(2), 164-180.

Borsotti, V. (2018, May). Sigsoft distinguished paper - Barriers to gender diversity in software development education: Actionable insights from a Danish case study. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)* (pp. 146-152). IEEE.

Bundsgaard, J., Bindslev, S., Caeli, E. N., Pettersson, M., & Rusmann, A. (2019). *Danske elevers teknologiforståelse: resultater fra ICILS-undersøgelsen 2018*. Aarhus Universitetsforlag.

Cohoon, J. P., & Tychonievich, L. A. (2011, March). Analysis of a CS1 approach for attracting diverse and inexperienced students to computing majors. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 165-170).

Daniel, B. (2015). Big Data and analytics in higher education: Opportunities and challenges. *British journal of educational technology*, *46*(5), 904-920.

DEA & Microsoft (2019) Hvordan får vi STEM på lystavlen hos børn og unge? - Og hvilken rolle spiller køn for

interesseskabelsen? https://www.datocms-assets.com/22590/1605692412-deastem-rapport-endelig.pdf

Eickelmann, B. (2019). Measuring secondary school students' competence in computational thinking in ICILS 2018—challenges, concepts, and potential implications for school systems around the world. In *Computational thinking education* (pp. 53-64). Springer, Singapore.

Fisher, A., & Margolis, J. (2003, January). Unlocking the clubhouse: Women in computing. In *Proceedings of the 34th SIGCSE technical symposium on Computer science education* (p. 23).

Grabarczyk, P., Nicolajsen, S. M., & Brabrand, C. (2022, November). On the Effect of Onboarding Computing Students without Programming-Confidence or-Experience. In *Koli Calling'22: 22nd Koli Calling International Conference on Computing Education Research* (pp. 1-8).

Jenkins, T., & Davy, J. (2002). Diversity and motivation in introductory programming. *Innovation in Teaching and Learning in Information and Computer Sciences*, *1*(1), 1-9.

Marcher, M. H., Christensen, I. M., Grabarczyk, P., Graversen, T., & Brabrand, C. (2021, August). Computing Educational Activities Involving People Rather Than Things Appeal More to Women (CS1 Appeal Perspective). In *Proceedings of the 17th ACM Conference on International Computing Education Research* (pp. 145-156).

Miles, M. B., Huberman, A. M., & Saldaña, J. (2014). *Qualitative data analysis: A methods sourcebook.* 3rd.

Mishkin, A. (2019, February). Applying self-determination theory towards motivating young women in computer science. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 1025-1031).

Niemiec, C. P., & Ryan, R. M. (2009). Autonomy, competence, and relatedness in the classroom: Applying self-determination theory to educational practice. *Theory and research in Education*, *7*(2), 133-144.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, *13*(2), 137-172.

Tomkin, J. H., & Charlevoix, D. (2014, March). Do professors matter? Using an a/b test to evaluate the impact of instructor involvement on MOOC student outcomes. In *Proceedings of the first ACM conference on Learning@ scale conference* (pp. 71-78).