Variables Affecting Students' Success in CS2

Camilla Björn
KTH Royal Institute of Technology
Stockholm, Sweden
cabjorn@kth.se

Viggo Kann
KTH Royal Institute of Technology
Stockholm, Sweden
viggo@kth.se

ABSTRACT

When trying to understand student success in computer science, much of the attention has been focused on CS1, leaving follow-up courses such as CS2 less researched. Prior studies of CS2 have often taken a deductive approach by focusing on predetermined variables such as CS1 grades, the impact of different paths from CS1 to CS2, gender and race. Although this has resulted in a better insight into these variables, we wonder if there might be another way of viewing which variables affect the students' success in the course. We have therefore chosen an inductive approach to better understand what these variables might be and how they interplay. This was done by analysing 16 semi-structured interviews with students enrolled in CS2 who have another speciality than computer science. The interviews focused mainly on the students' methods for succeeding in the course, experiences of the course and programming background. Through a thematic analysis of the interviews, we found the following five main success variables for CS2: programming competence, computer literacy, opportunity to receive help, ability to help oneself and teaching. These variables can in several cases be related to the ones previously addressed, however, they can also offer a different perspective on student success in the course.

CCS CONCEPTS

• **Social and professional topics → Computer science education**.

KEYWORDS

Student Performance, CS2, Programming Competence, Computer Literacy, Help

ACM Reference Format:
Camilla Björn and Viggo Kann. 2023. Variables Affecting Students' Success in CS2. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2023), July 8–12, 2023, Turku, Finland.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3587102.3588856

1 INTRODUCTION

Courses in computer science are common additions to many degree programs, not only those specialising in computer science or IT. Many programs only include a course in basic programming (CS1), however, a large number also include a second course focusing on algorithms and data structures (CS2). Contrary to CS1, which rarely requires any prerequisites in computer science, CS2 often requires completion of CS1 to allow the students to use programming as a tool when learning about the algorithms and data structures covered in the course. Compared to CS2, CS1 has been subject to extensive research. It has even earned its own ACM CCS (Computing Classification System) concept code and there has been a large interest in predicting and understanding students' success in the course. For example, some of the earlier research discovered that comfort level in the class, competence in mathematics and programming behaviour are important factors for success in the course [21, 22].

As the popularity of computer science grows [8], more and more students with varying backgrounds will naturally enter the field. The number of students who are already able to program when entering higher education is also likely to increase, as more effort is placed on programming in K-12 education [11]. In Sweden, for example, programming was added to the compulsory curriculum in 2017 through inclusion in the mathematics education [15]. The students who have spent more time in the Swedish compulsory school system after 2017 are therefore likely to have developed substantial programming competencies, possibly surpassing those required in CS1. A better understanding of what makes students succeed in their education is always important. However, one could argue that it is particularly important in larger introductory courses since more students are affected and a larger number of them might not even have chosen computer science as their major.

While CS1 has received a large amount of attention, considerably less has been placed on factors that could affect student performance in CS2. However, that is not to say that no work has been done. Previous research has shown several factors that play a significant role in students' performance in CS2, such as success in CS1, prior programming experience, success in mathematics and different pathways from CS1 to CS2 [3, 10, 14, 17]. While these studies have been successful and contributed with important results, their deductive nature limits the results to areas that are already known or hypothesised since the factors they investigate are predetermined. It is possible that there are several other factors that also play a significant role in students' success in CS2 which may never be considered. In this paper, we will therefore take on an inductive approach to explore if there might be additional ways of viewing which variables affect students' success in the course. To allow for a deeper understanding, we will explore the matter qualitatively, hence we are not aiming to state the significance of our findings. We believe this is an important starting point for a richer understanding of students' success in CS2.

Our study will therefore be guided by the two research questions, presented in section 2 below.

ITiCSE 2023, July 8–12, 2023, Turku, Finland

ACM ISBN 979-8-4007-0138-2/23/07.
https://doi.org/10.1145/3587102.3588856

2 RESEARCH QUESTIONS

These research questions are limited to the context of our work. The first question should be understood as focusing on the variables present in our material and is unlikely to produce a conclusive picture of all variables affecting students' success in CS1.

RQ 1 Which variables affect the students' success in CS2?
RQ 2 How do these variables interplay with each other?

3 RELATED WORK

Computer science education researchers have been interested in examining factors influencing students' success in computer science courses and predicting student performance since at least the early 1970s [1]. This is an important area of research since the ability to understand student success enables opportunities to improve educational outcomes by helping educators allocate resources and instruction more accurately [12].

In 2008, an ITiSCE working group conducted a systematic literature review of work predicting students' academic success [12]. The review was limited to publications using quantifiable predictors of success in computer science, engineering, and informatics. In addition to identifying predictors, the group also analysed metrics used to define success. They found individual course grades to be the most commonly used metric, followed by assessment or assignment performance, course or program retention and overall GPA. Out of the 29 performance predictors that they found, course performance, course engagement and performance in previous courses stood out as the most common. They also noted that some aspects of demographics data, such as gender and health, had become increasingly more common, despite predictors from this category otherwise rarely being used. Other increasingly more common predictors were psychometric factors, such as self-regulation and self-efficacy.

As noted earlier, the majority of the work related to identifying factors indicating students' success in computer science education has been performed on CS1, however, there is still work focused on CS2. In alignment with the ITiCSE working group literature review [12], a large number of the studies used course grade [3, 10, 14, 17, 20] as the metric for success, but exam score has also been used [2]. As noted in the literature review [12], one of the most frequently examined factors for indicating success in computer science education courses was the performance in previous courses. This also appears to be the case in the literature examining factors for success in CS2. In 2022, Hooshangi et al. [14] concluded that CS1 grades are significant factors in indicating students' CS2 grades, which is consistent with other studies [10, 17]. They also found that neither gender nor race were significant factors in students' success. A particularly interesting result from this study was that the students enrolled in competitive majors performed significantly better whereas students who had transferred CS1 credits or had been enrolled in more than one other CS course before CS2 had a significantly higher failure rate. Bisgin et al. [3] also examined the impact performance that previous courses had on the success in CS2 and found a significant correlation with calculus and object-oriented programming. In addition, they also found a significant correlation between the students' perception of difficulty in the course and their success.

4 COURSE CONTEXT

The CS2 course studied in this paper is given to engineering students who have another area of specialisation in their bachelor than computer science at KTH Royal Institute of Technology in Sweden. These students are all enrolled in 5-year combined bachelor and master programs. Many of them are required to complete the course as a mandatory part of their program or in order to select a specific master's specialisation. However, some of the students choose it as an elective course. Due to the large number of students having to complete the course, it is given every semester, and the cohort naturally consists of students from many different study programs. In 2022, 271 students were enrolled in the spring offering of the course and 222 in the autumn.

Before enrolling in CS2, all students must have completed the introductory course in programming, CS1, at KTH or an equivalent course at another university to ensure that they are familiar with basic object-oriented programming in Python. This is the only required prerequisite for the course.

The time span between CS1 and CS2 can vary between different study programs and between different students. Some students only have a summer or winter break between the two courses, whereas many others have up to a year and some even close to two years. There are also a few students who have an even longer break between the courses due to choosing CS2 as an elective course in their final year or having fallen behind in their study plans.

The focus of the CS2 course is algorithms and data structures, but it also contains some additional program development content such as program quality, abstraction, modularisation, testing, system calls and standard modules. Although the content of the course is mostly theoretical, there are still many practical components, since the students have to implement many of the algorithms and data structures as well as use them to solve problems in Python. [16]

Since the course has both theoretical and practical learning objectives, the assessment is divided into two modules, a theoretical module and a lab module, which are assessed and graded separately. The final course grade is then calculated based on the mean of the grades from the two modules. A passing grade in the theoretical module is obtained by passing five biweekly mini-exams. For a higher grade, they also need to pass one to two additional oral exams, depending on the grade. The practical module consists of 10 mandatory programming lab assignments which the students present weekly to a teaching assistant (TA). The students are encouraged to work in pairs but allowed to work alone if they prefer. To obtain a higher grade in the lab module they also need to complete one to two additional lab assignments of greater complexity, depending on the grade.

5 METHODOLOGY

In order to achieve a better understanding of variables affecting students' success, we have chosen to use semi-structured interviews which were transcribed and analysed thematically. In this section, we will describe more in-depth how the interviews and the analysis were conducted and the rationale behind our approach.

5.1 Data Collection

During the two CS2 course offerings in 2022, the first author conducted a total of 16 semi-structured interviews. The first seven were part of a pre-study [4] during the spring semester and the remaining nine complementary interviews were conducted in the following course offering during the autumn semester. In the interviews, the students were first asked about their programming background such as experience in and perceived proficiency after CS1 and potential additional programming experiences. They were then asked extensively about their experience of CS2, their strategies when working with the lab assignments and preparing for the mini-exams and potential difficulties that they have had in the course and how they overcame them. The overall structure of the interview guide remained roughly the same during the two rounds, however, some additional follow-up questions were added together with a set of questions about computer literacy. The additional focus on computer literacy was motivated by the spring interviews in which several students had brought up difficulties in the course which we classified as poor computer literacy rather than poor programming competence.

Before the first round of interviews, an invitation was sent out to all course participants along with a short recruitment pitch during a lecture break. This resulted in interviews with students with a relatively wide range of experiences of different aspects of the course. Since we assumed that the students might have struggled more in the earlier stages of the course and we wanted them to be able to remember these potential difficulties while at the same time have had some time to overcome them, these interviews were conducted rather early in the course (during the weeks of lab assignment 4 and 5). For the second round of interviews, we wanted to give the students more time to overcome potential difficulties and gather experiences from the course. Hence these interviews were conducted during the week of lab assignment 7 and in two cases lab assignment 9. However, the recruitment process for these interviews unfortunately yielded a more homogeneous group of students. We started recruiting students through a short pitch during a lecture break. This resulted in seven students signing up, however during the interviews, it became clear that these students had had a relatively smooth path through the course. In an attempt to reach out to students with more varying experiences of the course, we sent out an email to all students who had at least one unfinished lab assignment. This resulted in two additional students signing up. All students who participated were offered a cinema ticket as a sign of gratitude for their time commitment and generosity in sharing their experiences. In total the students represented seven different study programs (three programs from the spring interviews and five from the autumn interviews) and six of the 16 interviewees were women (three women in each round of interviews). The interviews were between 17 to 46 minutes long, however, the average interview was 31 minutes.

Due to the nature of the information that we would gather during the interviews, Swedish law did not require us to apply for ethical approval [9]. However, we did take several ethical considerations into account before starting the data collection. All students participating in the interviews were assured that the interviewer, who also worked as a teaching assistant in the course, would not participate during their assessment, to ensure that their course assessment would not be affected by their participation. They were also informed that they could withdraw their consent without having to state a reason both during and after the interview and that the transcripts would be anonymised and only analysed by the authors.

5.2 Analysis

The interview data went through two cycles of analysis. The initial cycle was performed on the first seven interviews to improve the interview protocol for the second round of interviews the following semester. The second cycle was conducted on all 16 interviews and was based on Braun and Clarke's guidelines for inductive reflexive thematic analysis [6, 7]. We chose this approach of thematic analysis since it is data-driven, meaning that the themes which we arrived at were derived from the interview data rather than a predetermined theoretical framework. Part of the beauty of reflexive thematic analysis is its flexibility which allowed us to analyse the data in the way that we saw most suitable without being constrained to follow guidelines of stricter research methods. As advised by Braun and Clarke [6] we started by familiarising ourselves with the data. Technically, this process began already when the first author transcribed the interviews verbatim, but we started together by reading and discussing the outcomes of the transcripts. After this initial phase, the first author coded the interviews. The codes were primarily focused on the semantic meaning of the participants' statements. However, there were a few occasions where we considered the latent meaning, in particular when the participants discussed the consequences of their study techniques. The coding phase concluded with an initial set of themes which we discussed and revised through several iterations.

It is worth noting that success in this study is not determined by the students' grades, since the grade is limited by both the quality and accuracy of the assessments and it does not indicate success at different stages or in different aspects of the course. York, Gibson and Rankin define academic success using six different categories, where grade only is included in one of them [23]. In order to answer our research questions we instead assessed the students' success based on their descriptions of their experiences and their described success in different areas.

6 RESULTS

In this section, we present the results from the analysis, starting with the first research question. When appropriate, excerpts from the interviews are included to motivate and better explain the results. All names presented are pseudonyms to preserve the anonymity of the students. Furthermore, the pseudonyms are gender-neutral, since we do not want to risk that the examples are misinterpreted as gender specific. They are merely there to illustrate important aspects of the success variables.

6.1 Success Variables

When analysing the interview data, we found five major variables which appear to affect the students' success in CS2. These five variables were: *programming competence*, *computer literacy*, *opportunity to receive help*, *ability to help oneself* and *teaching*. Below

we present each variable and give at least one example from an interview where the variable appeared.

6.1.1 Programming Competence. This is the most obvious success variable since the only official prerequisite for CS2 at our university is CS1, in which the students are expected to develop a foundation in object-oriented programming in Python. Since the lab assignments require the students to be able to implement the theoretical components of the course, students who were lacking in programming competence found these assignments more difficult as explained by Elliot:

> "The thing that's been the hardest in CS2 has probably been the actual programming. Maybe not to understand the theoretical computer science parts, but like... we just had a lab where we were going to replace all letters in a word systematically, damn that was hard."

On the other hand, students with greater competence in programming did not express the lab assignments to be unreasonably difficult. However, they sometimes emphasised, like Lee below, that it was the theoretical aspects of the course that were new to them.

> "I'm not that used to these more abstract things [...] but it gets better when you practise it. But the actual Python syntax and stuff like that, that's no problem."

6.1.2 Computer Literacy. During the first round of interviews, it became clear that some students encountered computer literacy related difficulties in the course. These issues could be downloading course material, locating files in their computers or installing an IDE or the right version of Python. These are additional skills that the students may need, that are never officially taught in the course or in the prerequisite course. Below, Kim provides an example of a computer literacy related problem:

> "I actually had the same problem with Python before, just before we were going to start with lab 1. [...] It's been the same problem with Python several times. [...] I know it has to do with how my files are stored, but I don't really understand it... how to fix it."

6.1.3 Opportunity to Receive Help. Several of the students had encountered difficulties during the course which they solved by finding someone who was able to help them. This could either be someone who was working in the course, such as the teaching assistants (TAs) or the lecturer, or it could be friends or family. All students enrolled in the course were offered help by the TAs during specific help sessions which were heavily utilised by some, such as Charlie:

> "We have spent a lot of time in the help queue and gotten help from there [from the TAs]. Well, we have also tried a bit during the lectures by asking the teacher."

Although all students were offered help during the course, some may for different reasons not feel comfortable asking a TA or the lecturer. One such example was given by Kim during the autumn interviews:

> "It feels more comfortable to ask friends [for help]. It feels like you have to know so much more when you're talking to someone who is a TA or something. Like if they... Not that they judge you... Well, but only a little. You're a little scared of that, maybe."

6.1.4 Ability to Help Oneself. It is natural to encounter difficulties and problems when learning something new, and one way of dealing with this is to find a way to help yourself without having to turn to others. A large number of the students mentioned that they solved many of their problems by searching for information online, such as Billie below, and some also mentioned reading the course material.

> "I also learned that Google is your best friend. That thing, StackOverflow, has a lot of good information, so it was never really that hard and I had a fairly easy time learning everything I had to learn."

6.1.5 Teaching. The students who signed up for the interviews were overall satisfied with the quality of the teaching in the course. Many claimed that it helped them learn the subject well, which in turn helped them understand how to complete the lab assignments. Robin, for example, explained how the lectures prepared them for the assignments:

> "Well, it kind of explains step by step how the different systems and algorithms work in a way that becomes pretty easy to transform to your own and apply to these labs."

If the students would have struggled with the methods used in the teaching, they would likely have had a more difficult time progressing in the course. Robin also gave an example of how they were negatively affected by the teaching when they struggled with a topic in one of the mini-exams.

> "It was a mini-exam that we had about compression [...] we had had a lecture before by a substitute lecturer that didn't really match the notes on Canvas [the online learning platform]. And then, when we had the mini-exam, there was a lot that I didn't think correlated between my perception and the material."

6.2 Interplay Between the Variables

In this section, we emphasise the impact of the five *success variables* by noting that students can have access to varying levels of each of them. Hence, this section is dedicated to the second research question which aims to exemplify the dependencies between the variables.

6.2.1 Programming Competence. The students with greater proficiency in programming often expressed less dependency on the other variables, with the exception of the *ability to help oneself* and/or *teaching*. Some of the students with greater programming competence mentioned that they valued the teaching highly and both attended lectures and appreciated the course structure. However, other students with greater programming competence were less affected by the *teaching* since they had proceeded to *help themselves* by learning the material on their own, such as Michele:

> "But I usually didn't go to programming lectures much before either and it has felt pretty easy to get started anyway."

Students with lower programming competence on the other hand relied more on other variables, such as the *opportunity to receive help*. They could also rely more on the *teaching* and the *ability to help oneself*.

6.2.2 Computer Literacy. This variable appears to relate to the other variables in a similar manner as *programming competence*. Many of the students who expressed high *programming competence* also expressed high *computer literacy*. However, it is possible for a student to have achieved high *programming competence* through CS1, without having great *computer literacy*, since this is never taught or assessed. These students could encounter technical difficulties in the course which in turn could affect their success. One such difficulty could for example be to read files. They might be able to write the code but struggle to find the whole path to the file if that is required, as expressed by Kim above.

6.2.3 Opportunity to Receive Help. Some students very clearly used this variable to compensate for low *programming competence* and/or *computer literacy*. They could also use this variable when they did not have the *ability to help themselves* as described by Taylor:

> "I don't usually google much. I mostly ask people because I think it is difficult sometimes to phrase the problem well in Google. But to friends, you can show and explain so that they understand [what you mean]."

This variable is not necessarily accessible to all students. Some students might not have friends in the course and if they are also uncomfortable asking the teacher or TAs for help, as Kim above, they have to rely more on the other variables which could be difficult if they have low *programming competence* and *ability to help themselves*.

6.2.4 Ability to Help Oneself. Students with great ability to help themselves should be able to compensate when their access to other variables is lower. Taylor for example, explained how they visited blogs to read different explanations of the same definition when the *teaching* was not enough:

> "It's a lot of googling. [...] Check out blogs that have like 5 readers, and then maybe someone has simplified it in a way that no one has explained it to you before and then it usually clicks."

6.2.5 Teaching. Well-structured *teaching* is unlikely to be enough to help the students with low *programming competence* since this is a prerequisite to the course and will not be the focus of the teaching. However, well-structured *teaching* is still an important variable and could help students with less access to other variables, such as for example poor *opportunity to receive help* since the *teaching* may lessen the need for help.

7 DISCUSSION

Our goal with this study is to provide deeper insight into different variables affecting students' success in CS2 by examining the topic inductively. The inductive approach allows us to study success in CS2 from a different angle than much of the previous work since the variables for success were never fixed at the start. This enables us to identify several success variables which have not, to our knowledge, previously been considered.

Earlier research has primarily examined quantifiable predictors of success [12], which is not the case with the variables identified in our study. To quantitatively study our variables, they would need to be associated with metrics. Out of our five variables, *programming competence* is the only one which has been extensively studied as a success factor in CS2. Works as [10, 14, 17], have all independently shown a significant correlation between success in CS1, the course in which the students start developing programming competence, and success in CS2, by using course grade as a metric. *Teaching*, has also been studied previously, however not in the context of being one of many factors indicating student success. For good reasons, there is no definition of what exactly constitutes good *teaching*. The number of possible decisions a teacher may take when designing their *teaching* is infinite, and there is no single best solution. To quantitatively investigate the effect of *teaching* on students' success, a specific teaching design has to be defined and tested on a cohort. An example of this is Hendrix et al.'s studio-based learning model, in which they measured the success through course grade [13]. Lewis and Massingill also present a teaching design, however, they attempted to measure the success through a self-assessment questionnaire [18].

The remaining three success variables: *computer literacy*, *opportunity to receive help* and *ability to help oneself* have, to our knowledge, not been the focus of research in this context. However, based on this study, we believe they play an important role in students' success in the course. Not only did we find examples of them in the interviews, but we also provide examples and/or motivations of how these variables interplay with each other. Based on our interviews, we for example found several students who, when struggling with the implementation of different theoretical concepts, described the *opportunity to receive help* as an important factor for their success. We also found that the majority of the students mentioned that they frequently used the internet to *help themselves* solve difficulties that they encountered in the course. One of the students did however point out that they much rather *received help* from friends since they found it difficult to formulate the search phrase.

During the interviews, we found indications that some students experienced computer-related difficulties in the course that were not related to their *programming competence*, but rather their *computer literacy*. The students who we interviewed had all been able to *receive help* to solve their issues. However, the examples highlight barriers in the course that could interfere with their success. Compared to *programming competence*, which they should obtain in CS1, the students may never have been trained in *computer literacy*, but yet, they may need it to succeed in CS2.

7.1 Limitations and Threats to Validity

No research method or approach comes without flaws and it is important to understand what they bring to the table. In this study, we have used a qualitative inductive approach which is suitable for deepening the understanding of the nature and interaction of variables which previously may have been unknown [5]. As emphasised by Black, qualitative research "seeks the answer to the

'what' question, not the 'how often' one" [5] which is true also for this study. We have chosen not to present any statistics related to our findings since calculations based on 16 participants would not be useful to the community. We want to minimise the risk of our results being inappropriately generalised and we have no intention of making claims which we do not have support for. This is also the reason why we have chosen not to present the gender, ethnicity and study program along with the quotes.

Although we have not investigated the significance of the variables highlighted in this study they were all present in our data. As mentioned in the methodology, we would have preferred a more even split between the interviewees who struggled in the course and the ones who did not. This would likely have contributed with richer descriptions of some of the variables and potentially also with additional variables that were not present in the current data. However, the lower number of interviewees struggling in the course is not necessarily an indicator that our findings are invalid. Potentially, quite the opposite since we were still able to find support for all variables presented. We for example still found support for computer literacy, despite this being a variable that might be stronger associated with students who are struggling. However, more research is of course needed to better understand these variables and their significance.

As noted above, this study can not contribute information regarding the significance of the variables in terms of how common they are. However, the results of the second research question do indicate that some variables might play a more significant role depending on the state of other variables. With that said, this study can not be used to determine if any of the variables have a greater effect on the students' success in the course.

Qualitative studies are often said not to be generalisable since the sample size is usually small. However, we would argue that this rather comes done to a question of what we are trying to generalise. When discussing generalisability in qualitative studies, Maxwell emphasises the importance of describing the context since this is crucial for understanding whether the results could be transferable to similar contexts [19]. This is, however, also the case with quantitative studies, since they too are context dependent. We would argue that the main difference is what we are trying to generalise. In our case, we found support for five different variables and we believe it is likely that these might be present in other CS2 courses in similar contexts.

8 CONCLUSIONS

Our goal with this paper is to provide a richer understanding of students' success in CS2 by using an inductive research approach. Through a qualitative thematic analysis of 16 semi-structured interviews with CS2 students, we identify the following five different success variables: *programming competence*, *computer literacy*, *opportunity to receive help*, *ability to help oneself* and *teaching*. We also provide examples of how these variables can interplay with each other. We hope that this work may inspire others to consider some of these variables which have received less attention, such as *computer literacy*, *opportunity to receive help* and *ability to help oneself*, in the research of students' success in CS2.

ACKNOWLEDGMENTS

We would like to thank the students who participated in the interviews for both their time commitment and contribution of insights. We would also like to thank the lecturers in the respective course offering for contributing with their time during the interviews in the pilot study as well as for helping us reach out to the students during the interview recruitment.

REFERENCES

[1] Carol Ann Alspaugh. 1972. Identification of some components of computer programming aptitude. *Journal for Research in Mathematics Education* 3, 2 (1972), 89–98.

[2] Leland Beck, Patty Kraft, and Alexander W. Chizhik. 2022. Predicting Student Success in CS2: A Study of CS1 Exam Questions. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1* (Providence, RI, USA) *(SIGCSE 2022)*. Association for Computing Machinery, New York, NY, USA, 140–146. https://doi.org/10.1145/3478431.3499276

[3] Halil Bisgin, Murali Mani, and Suleyman Uludag. 2018. Delineating Factors that Influence Student Performance in a Data Structures Course. In *2018 IEEE Frontiers in Education Conference (FIE)*. 1–9. https://doi.org/10.1109/FIE.2018.8659300

[4] Camilla Björn. 2022. Non-CS Students' Progression from CS1 to CS2. In *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 2* (Dublin, Ireland) *(ITiCSE '22)*. Association for Computing Machinery, New York, NY, USA, 612. https://doi.org/10.1145/3502717.3532139

[5] Nick Black. 1994. Why we need qualitative research. *Journal of epidemiology and community health* 48, 5 (1994), 425.

[6] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[7] Virginia Braun and Victoria Clarke. 2019. Reflecting on reflexive thematic analysis. *Qualitative research in sport, exercise and health* 11, 4 (2019), 589–597.

[8] Tracy Camp, W. Richards Adrion, Betsy Bizot, Susan Davidson, Mary Hall, Susanne Hambrusch, Ellen Walker, and Stuart Zweben. 2017. Generation CS: The Growth of Computer Science. *ACM Inroads* 8, 2 (may 2017), 44–50. https://doi.org/10.1145/3084362

[9] Swedish Research Council. 2017. *Good Research Practice*. Swedish Research Council, Stockholm.

[10] Holger Danielsiek and Jan Vahrenhold. 2016. Stay on These Roads: Potential Factors Indicating Students' Performance in a CS2 Course. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (Memphis, Tennessee, USA) *(SIGCSE '16)*. Association for Computing Machinery, New York, NY, USA, 12–17. https://doi.org/10.1145/2839509.2844591

[11] Varvara Garneli, Michail N. Giannakos, and Konstantinos Chorianopoulos. 2015. Computing education in K-12 schools: A review of the literature. In *2015 IEEE Global Engineering Education Conference (EDUCON)*. 543–551. https://doi.org/10.1109/EDUCON.2015.7096023

[12] Arto Hellas, Petri Ihantola, Andrew Petersen, Vangel V. Ajanovski, Mirela Gutica, Timo Hynninen, Antti Knutas, Juho Leinonen, Chris Messom, and Soohyun Nam Liao. 2018. Predicting Academic Performance: A Systematic Literature Review. In *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education* (Larnaca, Cyprus) *(ITiCSE 2018 Companion)*. Association for Computing Machinery, New York, NY, USA, 175–199. https://doi.org/10.1145/3293881.3295783

[13] Dean Hendrix, Lakshman Myneni, Hari Narayanan, and Margaret Ross. 2010. Implementing Studio-Based Learning in CS2. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. Association for Computing Machinery, New York, NY, USA, 505–509. https://doi.org/10.1145/1734263.1734433

[14] Sara Hooshangi, Margaret Ellis, and Stephen H. Edwards. 2022. Factors Influencing Student Performance and Persistence in CS2. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1* (Providence, RI, USA) *(SIGCSE 2022)*. Association for Computing Machinery, New York, NY, USA, 286–292. https://doi.org/10.1145/3478431.3499272

[15] Cecilia Kilhamn, Lennart Rolandsson, and Kajsa Bråting. 2021. Programmering i svensk skolmatematik: Programming in Swedish school mathematics. *LUMAT: International Journal on Math, Science and Technology Education* 9, 1 (2021), 283–312.

[16] KTH. 2021. *DD1320 Applied Computer Science 6.0 credits*. https://www.kth.se/student/kurser/kurs/DD1320?l=en

[17] Lucas Layman, Yang Song, and Curry Guinn. 2020. Toward Predicting Success and Failure in CS2: A Mixed-Method Analysis. In *Proceedings of the 2020 ACM Southeast Conference* (Tampa, FL, USA) *(ACM SE '20)*. Association for Computing Machinery, New York, NY, USA, 218–225. https://doi.org/10.1145/3374135.3385277

[18] Mark C. Lewis and Berna Massingill. 2006. Graphical Game Development in CS2: A Flexible Infrastructure for a Semester Long Project. In *Proceedings of the 37th*

SIGCSE Technical Symposium on Computer Science Education (Houston, Texas, USA) *(SIGCSE '06)*. Association for Computing Machinery, New York, NY, USA, 505–509. https://doi.org/10.1145/1121341.1121499

[19] Joseph A Maxwell. 2021. Why qualitative methods are necessary for generalization. *Qualitative Psychology* 8, 1 (2021), 111.

[20] Sander Valstar, William G. Griswold, and Leo Porter. 2019. The Relationship between Prerequisite Proficiency and Student Performance in an Upper-Division Computing Course. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. Association for Computing Machinery, New York, NY, USA, 794–800. https://doi.org/10.1145/3287324.3287419

[21] Christopher Watson, Frederick W.B. Li, and Jamie L. Godwin. 2014. No Tests Required: Comparing Traditional and Dynamic Predictors of Programming Success. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. Association for Computing Machinery, New York, NY, USA, 469–474. https://doi.org/10.1145/2538862.2538930

[22] Brenda Cantwell Wilson and Sharon Shrock. 2001. Contributing to Success in an Introductory Computer Science Course: A Study of Twelve Factors. In *Proceedings of the Thirty-Second SIGCSE Technical Symposium on Computer Science Education* (Charlotte, North Carolina, USA) *(SIGCSE '01)*. Association for Computing Machinery, New York, NY, USA, 184–188. https://doi.org/10.1145/364447.364581

[23] Travis T York, Charles Gibson, and Susan Rankin. 2015. Defining and measuring academic success. *Practical assessment, research, and evaluation* 20, 1 (2015), 5.