UiT The Arctic University of Norway

Faculty of Science and Technology
Department of Computer Science

Integration of programming in Norwegian schools

The effects of prior programming experience on students in a university-level programming course

Thomas Vatne Eide

INF-3981 Master's thesis in Computer Science Spring 2024

Abstract

Background: In 2020 a curriculum renewal in Norway integrated programming into multiple subjects at both elementary schools and upper secondary schools. This was done with the hopes of improving deep learning and introducing computational thinking to pupils attending the schools. Some criticism has been raised against the decision, with some declaring that this will hurt deep learning and that programming is better fit being its own subject.

Aim: The study aims to answer the following questions:

- How does prior programming experience from science and technology subjects in upper secondary school affect students who attend first year programming courses at university?

- What attitude does university level computer science students have towards the curriculum renewal?

Methods: A qualitative case study were performed using data collected through semi-structured interviews of first year computer science students. There were six students interviewed in total, with two students who had prior programming experience from math class, and four students who did not have earlier experience.

Results: All students with previous programming experience enjoyed having programming as part of their class and felt mastery, however there were mixed responses when it came to the competence of their teachers. The experienced students also seemed more confident explaining the concept of programming, and managed to use real life examples in their descriptions. Most students connected programming to math and were positive to it being integrated into existing subjects rather than having it as its own subject. When faced with code snippets, the experienced group handled it systematically by analysing the inputs and outputs of the functions before moving on the the function itself, in contrast to the inexperienced students who read the code line by line starting at the top.

Conclusion: Students are positive to having programming integrated into the school curriculum, and their motivations are not affected by the competence of their teachers. Students seem to mostly have their competence restricted to python and its syntax.

Preface

Five years of university have gone by in the blink of an eye, and now I'm sitting here writing the preface to my master's thesis. It feels surreal just writing it, but after spending an overwhelming amount of time studying the curriculum renewal in Norway, I'm ready to take a step back, breathe, and enjoy my last moments as a computer science student.

Firstly, I want to give a huge thank you to my supervisor Henrik Løvold, not only for the immensely helpful guidance throughout the semester, but also for our ramblings during meetings that helped me relax and think about other things outside of the thesis.

Thank you to the participants of the study. You made this project possible and provided me with both data, experience, and a lovely chat.

And of course, thank you to my friends and family. You have cheered me on all the way, while also giving me much needed breaks. Be it either by dragging me to the concert of my life, always being available on the phone for a chat, or gaming until the sun rises, you guys have been invaluable in this process.

Contents

List of Figures

1 Introduction

In a rapidly changing and evolving society, its important that education evolves with it. Today we are constantly surrounded by technology both privately, and in work spaces, no matter whether you are a nurse, mechanic, teacher, or biologist (Kunnskapsdepartementet, 2017, p. 3). In order for pupils to become active participants in a digital society, schools have to take into consideration the future of technology, and prepare pupils with future-oriented competence (NOU 2015:8, 2015, p. 7).

Several countries have already considered this and implemented curricula related to technology and programming, with Norway being no exception (Bocconi, Chioccariello, & Earp, 2018). After a multitude of studies appointed by the Norwegian government, a curriculum reform was enacted which would give pupils skills for the jobs of tomorrow (Utdanningsdirektoratet, 2021a).

1.1 The curriculum renewal

In 2020, the Norwegian government set a new precedent for education in Norway, by updating their National Curriculum. This new curriculum is called "kunnskapsløftet 2020", or K20, and focuses on a number of forward-leaning factors that should make pupils more prepared for the future.(Utdanningsdirektoratet, 2021a) Some of the most important points denoted by the curriculum are "deep learning", in which pupils are to learn the core topics of a subject well, "learning

to learn", a method of teaching such that pupils can keep learning independently and "computational thinking and programming", which focuses on using algorithms in order to solve problems, both with and without computers. These three aspects have all been identified as important through a number of committees appointed by the Norwegian government, all tasked with reviewing education through the lens of the future (NOU 2013:2, 2013; NOU 2015:8, 2015; NOU 2015:15, 2015). As part of this renewal, programming as been integrated into multiple subjects in school, primarily mathematics. This decision has been met with some criticism, with some proclaiming that it won't fit the already established traditions of existing subjects(Sanne et al., 2016, p. 76).

1.2 The goal of the study

This project is in the form of a case study, and revolves around the students having experienced the curriculum renewal. In the core of the project is a single research question:

- How does prior programming experience from science and technology subjects in upper secondary school affect students who attend first year programming courses at university?

During the course of the study, another aspect to consider presented itself through the question:

- What attitude does university level computer science students have towards the curriculum renewal?

As 2023 was the first year in which students who have experienced the curriculum renewal entered university, there are very few studies examining the students themselves. Studies that do exist show a positive trend in programming competence after the renewal, however more research is needed in order to come to a conclusion (Bolland, 2023a).

1.3 Motivations

Throughout my university degree, I've always been drawn to the noticeable difference in the way my fellow students and I have learned and understood programming. There was not a significant difference in skill levels, but a lot of the time there would be disagreements around what would be the best course of action when it came to the assignments. As for myself, I had no

prior experience with programming, and would often rely on my knowledge of mathematics in order to concoct a solution for the tasks we were posed with on a daily basis. To me, mathematics and programming both share the same aspect of problem solving, so by handling programming the same way i do mathematical problems, I managed to keep up with other students that had earlier programming experience.

Combining this with my experience teaching both at university as an assistant in mathematics, and at elementary schools, I became very curious as to the effects of programming in science, technology, engineering, and mathematics (STEM) subjects. Would this affect the way students think about programming, and what attitudes do the students themselves have towards programming?

1.4 Chapter explanations

Chapter 2 - Background. In depth explanation of the Norwegian curriculum renewal, and a look into other countries' curriculum concerning programming.

Chapter 3 - Methods. Discussing the choice of research methodology as well as how data collection has been performed.

Chapter 4 - Results. The results of the interviews are presented.

Chapter 5 - Discussion. Discussion around the results of the interviews, with a focus on answering the research questions.

Chapter 6 - Conclusion.

2

Background

2.1 Curriculum renewal

In order to keep up with the rapid evolution of society, the Norwegian Directorate for Education and Training enacted new curricula for Norwegian elementary and upper secondary schools in 2020 (Utdanningsdirektoratet, 2021a). This new curricula would take into account the emergence of future job spaces that do not exist yet, and focus on preparing the pupils with relevant and necessary skills. Of these skills, there are three that are especially relevant to our project at hand:

- Deep learning (Dybdelæring)
- Computational thinking (Algoritmisk tenking)
- Programming (programmering)

2.1.1 Deep learning

In the new curricula, the school subjects are shaved down to their essence and only describe the most important teachings for the students. This is in order to promote "Deep learning" (Utdanningsdirektoratet, 2021a). The direct definition given by Utdanningsdirektoratet (2019b) is "... det å gradvis utvikle kunnskap og varig forståelse av begreper, metoder og sammenhenger i fag og mellom

fagområder" (... to gradually develop knowledge and lasting understanding of terms, methods and contexts in subjects and between subject areas). This is streamlining teaching in such a way where more time can be used on learning precise material well, rather than having to spend less time on multiple coarse subjects that may confuse the pupils.

Deep learning also enables pupils to reflect over their own learning and make them independent in their search for knowledge. "Learning to learn" is a central principle in KL20, and the introduction of deep learning is a way of strengthening this principle (Regjeringen, 2017, p. 12). By creating their own questions and discovering new ways of arriving at answers, the pupil will get insight into how knowledge stretches across multiple subjects and create a basis for their lifelong learning.

2.1.2 Computational thinking

Another focus of KL20 is computational thinking. This is a way of problem solving that is reminiscent of the way a computer solves and performs tasks. Here, a problem is solved systematically by analysing what the task at hand is, while also suggesting a collection of possible solutions. (Utdanningsdirektoratet, 2019a)

UDIR has created a poster named "Den algoritmiske tenkeren" (The computational thinker) seen in figure 2.1 that depicts the key concepts and methods of computational thinking (Utdanningsdirektoratet, 2019a). Going step-wise through the concepts, computational thinking begins with analysing and predicting what the task at hand is, before creating a fitting approach or algorithm that may solve the task. The next step would then be to deconstruct the task into smaller, more manageable problems, which may reveal patterns and similarities to other solved problems. In order to focus on what is relevant to the task, it would then be fitting to strip the task of unnecessary details. Lastly the task can be evaluated and generalized such that it can be used to solve other similar problems.

The methods of figure 2.1 reflects qualities that a computational thinker should possess. Though computational thinking should be systematic, its also important to be creative, curious and receiving of all kinds of solutions. One should be critical such that errors are quickly picked up on, and should possess both determination and perseverance as to not give up on a task. Cooperation and sharing is also necessary for a successful computational thinker. (Utdanningsdirektoratet, 2019a)

Figure 2.1: "Den algoritmiske tenkeren" (The computational thinker) Source: (Utdanningsdirektoratet, 2019a)

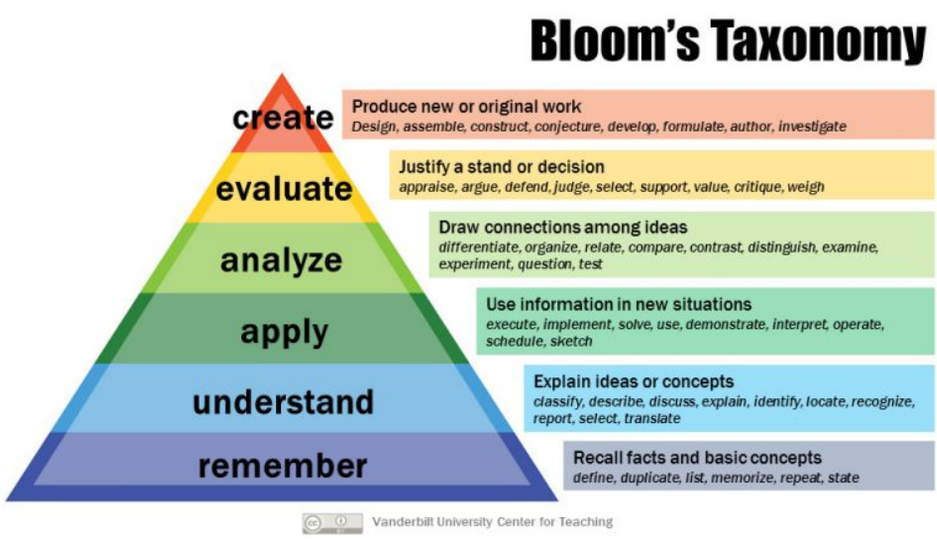

Figure 2.2: "Blooms taxonomy" Source: (Armstrong, 2010)

These qualities associated with the computational thinker does bear resemblance to the upper levels of a model named Bloom's Taxonomy (Armstrong,

2010). This model, as seen on figure 2.2, creates a hierarchical structure of skills and abilities for students with knowledge as the basis. While the lower levels of the model shows simple abilities such as remembering and understanding ideas, the higher levels involves applying, analyzing and evaluating the ideas, followed by creating new work based on those ideas. These abilities are what computational thinking and deep learning are made to strengthen.

2.1.3 Programming

What may be the most essential part of the new curricula related to this project is the introduction of programming. Utdanningsdirektoratet (2021a) declares programming a key competence for the future, and has adapted it into multiple different subjects both in elementary schools and upper secondary school. The following list portrays what subjects in Norwegian school has been updated to include coding as part of its curricula (Kunnskapsdepartementet, 2019a, 2019b, 2019c, 2019d; Vogt, 2021):

- Elementary school
  - Arts and crafts
  - Mathematics
  - Music
  - Natural sciences
- Upper secondary school
  - Mathematics
  - Physics
  - Biology

The degree to how much programming is involved in the subjects varies, with mathematics having the most learning objectives stretched across multiple grades (Kunnskapsdepartementet, 2019b). Having programming in school will also help develop the aforementioned "computational thinking", as this is a suitable problem solving technique to use in programming assignments. (Statped, 2021)

2.2 How the curriculum renewal came to be

The renewal was not a baseless act, and was put into action as a response to a multitude of studies performed beforehand. These studies will be described in the following section.

2.2.1 Digiutvalget

In 2011 a committee named the Digitization Commission was assigned to identify and address struggles related to digital innovation in Norway (NOU 2013:2, p. 3). There were a multitude of concerns identified by the committee, one of which being that the digital competence was low, which brought risks around aspects such as information security.

Even though technology was a field of priority in education, the committee stated that more effort has to be made in order for students to improve their digital skills (NOU 2013:2, p. 10). They saw a lacking competence in creating technology, knowing how programs and networks work, understanding computers in general, and programming. As the Digitization commission wants this competence to expand, they suggested adding programming as an elective subject in secondary school, and that readily available courses in programming should be held for children outside of school. Waiting until upper secondary school to learn programming is too late according to the committee, as this will result in pupils becoming consumers of technology, rather than producers. The committee also mentions that STEM subjects should be strengthened in general, as they are important in understanding programming, especially mathematics. (NOU 2013:2, p. 105)

2.2.2 Mathematics in Norwegian schools

In order to gain a full understanding of how mathematics is taught, a group in 2013 was appointed to review math as a subject from the first grade in elementary school to the last grade in upper secondary school (Borge et al., 2014, p. 1). A point brought up by the group was how mathematics is changing because of increased access to new digital tools. Because of this, new aspects of mathematics are more relevant than before, such as numerical methods and simple programming. They therefore concluded that the mathematics curriculum in upper secondary school should be reevaluated with more focus on digital tools (Borge et al., 2014, p. 85).

2.2.3 Ludvigsenutvalget

As the Norwegian government wanted to know what competence is necessary in the careers of the future, they designated a committee to examine this (NOU 2015:8, p. 3). Concepts such as "learning to learn" and "deep learning" are highlighted in their report, as they are both closely related to the development of competence. Both helps students understand what they have learned, how they can use it, and when they should use it, all attributes important to obtain

competence.

Digital competence is also made note of in the report, with the committee saying that its crucial when it comes to innovation and technical advancements (NOU 2015:8, p. 26). They also describe digital competence as suitable across multiple subjects, with examples such as information security and competence using digital tools. However, they suggest that in order to obtain a full digital competence and concentrate the teaching responsibility, it should be integrated into one or a few subjects. This competence will also help develop critical thinking skills, as well as communication and cooperation. As such, the committee concludes that digital competence should be learned on the same level as reading, writing and oral communication (NOU 2015:8, p. 47).

2.2.4 Lysneutvalget

A committee designated to digital weaknesses in 2014 built further unto the ideas presented by The Digitization Commission concerning information security (NOU 2015:15, p. 4). While The Digitization Commission wants programming as an elective subject in elementary school because of the desire to create producers rather than strictly consumers, the Lysne-committee wants this because of security reasons. More technical insight is needed to understand the possibilities and risks using digital assistance, and this insight will also be necessary in order to develop and improve technology as we know it (NOU 2015:15, p. 225). Other reasons for introducing programming into education is its great impact on technological innovation and economy, its assistance in learning other subjects, and its development of critical thinking and creativity.

2.2.5 Parliament report nr. 28

A report given by the Norwegian Parliament in 2016 was used as the basis when the new curricula were under development (Kunnskapsdepartementet, 2016; Utdanningsdirektoratet, 2021b). This report makes great use of the Ludvigsen Committee's findings to describe how the curricula in schools should be renewed for the future (Kunnskapsdepartementet, 2016, p. 8-9). For instance, the report agrees that there should be a more distinct separation of which subjects should teach digital competence (Kunnskapsdepartementet, 2016, p. 32). The importance of deep learning is also agreed upon by the paper (Kunnskapsdepartementet, 2016, p. 33). In order to improve digital competence, the paper wishes that the opportunity for programming in education should be built upon, and announces that tests having programming as an elective subject in secondary school will be held as of fall 2016 (Kunnskapsdepartementet, 2016, p. 32, 54)

2.2.6 Sanneutvalget

In 2016 a group was assigned to conduct a review of technology in education, and give thoughts on changes to education pertaining to technology and programming (Sanne et al., 2016, p. 3). The committee states that in order to obtain competence in how technology advances, it is crucial that the students themselves get to program. This will not only help their problem solving skills, but also give them an understanding of how technology is developed (Sanne et al., 2016, p.75). Like the previous studies discussed, this committee is also critical to how schools at the time primarily teached digital technology in such a way where the students becomes a consumer, rather than a producer and a creator. "The goal must be to give them (students) a foundational competence so that they can take control over the technology and creatively develop it for their own benefit as well as for society as a whole" ("Målet må være å gi dem et grunnlag av kompetanse slik at de selv kan ta kontroll over teknologien og kreativt utvikle den til sitt eget og fellesskapets beste.") (Sanne et al., 2016, p.76).

The committee is also against the thought of integrating technology and programming into existing subjects, and believes it should stand on its own in the curricula. They point to experiences both in Norway and other countries where technology aspects in other subjects gets neglected due to the subject having already established traditions (Sanne et al., 2016, p. 76). As a result of this, they suggest creating a new mandatory subject in elementary school that encompasses both programming and technology. Not only would this give every student the same opportunity of learning programming, which could help balance out the gender gap in STEM subjects, but it would also help renew other subjects to include more technology based applications. The committee uses mathematics as an example and describes how creative algorithm development could be introduced in mathematics, only if the students have programming experience beforehand (Sanne et al., 2016, p. 91).

2.2.7 Digitaliseringsstrategien

A Digitization strategy was published by the Norwegian government in 2017 concerning education in Norway. By making schools use more digital equipment, the strategy was created to better equip students for the future (Kunnskapsdepartementet, 2017, p. 3). Having students break down problems and solve it systematically like a computer is mentioned by the strategy, saying that there has been an increasing demand for learning computational thinking. With this in mind, the strategy states that the curriculum renewal will review how programming can be integrated into established subjects, with a focus on mathematics and natural science (Kunnskapsdepartementet, 2017, p. 18). Other things to be done during

the strategy period are: Enact programming as an elective subject as of 2019, enact programming and modelling as a specialization subject in upper secondary school as a national experiment as of fall 2018, and create a universal design for the use of digital learning materials in education (Kunnskapsdepartementet, 2017, p. 20).

2.3 Programming curricula in different countries

In order to get a fully fledged understanding of the subject at hand, its of utmost importance to review how other countries have chosen to handle technology and programming in schools, and what results they have obtained. The Sanne-committee made note of saying that technology as a subject in schools is relatively new for most countries, but also underlined how Norway is one of the few countries that haven't planned for programming to become its own subject (Sanne et al., 2016, p. 73). This section will first detail some previous research done in Norway after the curricula renewal was enacted, before detailing programming education in other countries.

2.3.1 Norway

As the curricula renewal is still quite fresh, there has not been an abundance of research done on the results of the renewal. There are however two studies that have happened since that give some indication on its effect.

A case study by Stenlund (2021) went into depths of the renewal through the eyes of the teachers. Through interviews with teachers, as well as observations during programming classes, the study concluded that the reception has been mixed. Though the teachers highlight programming and computational thinking as important skills, they also mention how difficult and time consuming it is to master programming both for the students and teachers. They are also quite critical to the renewal, especially elementary school teachers, as they feel they have received little time and support in order to prepare themselves for the addition of programming. There were also indications that teachers with prior programming experience were more critical to integrating programming in mathematics, and would rather see programming as its own subject (Stenlund, 2021, p. 82).

In 2023 a test named "The National Prior Knowledge Test in Programming" was conducted in order to evaluate the programming knowledge of students in higher education (Bolland, 2023a). In an extended paper, Bolland (2023b) explains that the test contained a series of programming tasks that addressed

programming concepts such as variables, loops and functions, with a maximum possible score of 22.60 points (Bolland, 2023b, p. 5). After some data pruning, results from 1 767 individuals was obtained which resulted in a collective score of 10.42, which is 46.1% of the maximum. However, when dividing the individuals into two groups, those who graduated from upper secondary school in 2023, and those that graduated before 2023, the results showed a clear difference. Those who graduated before 2023 and have therefore experienced the new curricula with programming obtained an average score percent of 63.3%, while the other group got an average percent of 39.4%. This seem to point towards an increase in programming competence having experienced the new curricula. Even so, Bolland (2023a) does point out how the results could also be affected by what classes the students participated in during upper secondary school. Removing students who chose programming as elective subjects, the average score for 2023 graduates drops to 54.47%. The score drops further if only examining the students who attended Practical Math 2, the most fundamental concluding maths course, with a score percent of 38.8%.

2.3.2 Sweden

In 2018, Sweden revised their curricula, with a focus on digitisation and developing digital competence in students (Skolverket, 2018, p. 8). The renewal promotes the use of digital tools, both by students and teachers, and has added digital tool competence into its general goals for pupils who finish compulsory schools (Skolverket, 2018, p. 12-12). Programming has also been integrated into both the subjects mathematics and technology. In mathematics, programming should be used in order to explore problems and make calculations, mostly through algebra (Skolverket, 2018, p. 55-56, 58-60). The goals in algebra mention constructing and using programming to create algorithms, and programming in different programming environments. In Technology, a more practical approach to programming is used, with the curriculum mentioning using programming in order to control objects (Skolverket, 2018, p. 297-299).

A study on a first year class with pupils aged 6-7 years was conducted in Sweden, where interviews, questionnaires and video footage were used in order to report on their experience with programming and computational thinking (Kjällander, Mannila, Åkerfeldt, & Heintz, 2021, p. 1). The lesson used in the study had pupils performing an coding assignment in ScratchJr, a block-based programming language suitable for young children. The study observed that the pupils had high self-efficacy when it came to programming, and expressed a positive attitude. Through a questionaire, about four of five pupils find programming fun, about half think they are skilled at programming, and more than half wants more programming. The students also cooperated and communicated with each other to a high degree during the lesson. This was in

contrast to the teachers of the class that claimed that they were uncomfortable and inexperienced with teaching programming (Kjällander et al., 2021, p. 6-7, 11-13).

2.3.3 Denmark

Denmark does not have any word covering computational thinking, and the ministry of education does not refer to the concept of computational thinking in any of its policy documents. "Teknologiforståelse" (understanding of technology) used in elementary schools, and "Informatik" (Informatics) used in upper secondary schools would be the two closest terms (Bocconi et al., 2018, p. 7). Digital competence and computational thinking is developed mainly through "IT and media", a cross curricular theme that is integrated in all subjects at elementary schools (Bocconi et al., 2018, p. 11).

In 2023, the Danish government presented a digitization strategy for all levels of education. The main motivation behind this strategy is the demand for workers with digital competence and understanding of technology (Uddannelses- og Forskningsministeriet, 2023, p. 8). A notable aspect of the strategy is how the government wants to add “Technology Understanding” as an elective subject in elementary schools, the first time in 30 years that Denmark has added a subject to elementary schools. Technology comprehension is also to be further developed in danish teacher education (Uddannelses- og Forskningsministeriet, 2023, p. 14).

2.3.4 Finland

In 2014, Finland revised their curriculum in order to include computational thinking and programming. This revision was implemented in 2016 (Bocconi et al., 2018, p. 13). Though "coding" is not mentioned in the Finnish school curriculum, it does explicitly mention computational thinking whenever programming is discussed (Bocconi et al., 2018, p. 8). Programming is integrated mainly into maths and crafts, but is also part of digital competence which stretches across all subjects. Similarly to Sweden, programming in maths covers creation of simple program in order to solve problems, while programming in crafts cover programming of physical objects such as robots (Bocconi et al., 2018, p. 18).

2.3.5 United Kingdom

The UK implemented the subject "Computing" in 2014, as a replacement to the existing subject, Information and Communication Technology (ICT) (Sentance & Csizmadia, 2017, p. 473). This is based on the computational thinking principle and aims to educate pupils in computational thinking and information technology, such that the pupil can navigate a digital world (Department for education, 2013). Not only should the pupil be digitally literate, but they should also be able to create programs and systems, and in general express themselves digitally.

2.3.6 United States

There is no nationally standardised curriculum in the United States, however a lot of teachers have chosen to adopt a curriculum based on the CSTA K-12 Computer Science Standards, a curriculum developed by Computer Science Teachers Association (CSTA) (Kusaka, 2021, p. 77). Though the curriculum does not use the term computational thinking, it defines five main aspects of computer science: Computing systems, networks and the internet, data and analysis, algorithms and programming and impacts of computing (Kusaka, 2021, p. 80).

3

Methods

As this project revolves around an open-ended question as well as human participants, the choice of methods could greatly impact the results obtained and the conclusions drawn. Therefore a great amount of time was spent choosing and justifying the methods.

This section will go into detail as to what kind of research this project is portraying, what methods were chosen and why, and describe the consequences of the chosen methods. A subsection will also be dedicated to discussing the ethics of the project, and the process of confirming that the research follows the correct procedures for working with personal details. Lastly, the validity of the data will also be discussed.

3.1 Choice of research methodology

In our research, the first step was to determine what research methodology to choose for this project. Though research projects often consist of mixed methodologies, they can usually be boiled down to qualitative or quantitative research. It's therefore important to have a clear understanding of what these two approaches entail, and how they differ.

3.1.1 Quantitative research

Quantitative research uses numbers and statistics in order to conduct the research (Hancock, Algozzine, & Lim, 2021, p. 5). An example of such research would be an end-of-year survey for a class at a university where the students grade their enjoyment of the class. Holliday (2007, p. 2-4) provides and reflects on a number of other examples such as surveys determining the population of a nationality with brown and blue eyes, and how many first-time buyers would buy a Ford rather than a Peugeot. These examples are used by Holliday to illustrate how simple questions often have a number of variables that may affect the result. For example the first-time buyers may have been influenced by societal factors such as acquaintances owning Fords or advertisements showing off Peugeots. Variables need to be controlled in order to create a result that can be replicated, and this will often be difficult when using quantitative research methods. This is not to say quantitative research is bad, as it's an excellent way of gaining data over a short time, since the data is usually obtained using simple instruments like questionnaires and surveys, as pointed out by Hancock et al. (2021, p. 7-8). They go on to explain that quantitative research is suitable if there is only a few variables that are to be investigated, such as "does online classes make students unhappy?" where a simple questionnaire answered by students could give an answer. Other suitable reasons provided by them are when either the participants or the conductor of the research has little time available, if the numbers of the results are more important than the words, and if the situation can comfortably be investigated from the researcher's perspective, rather than the participants.

3.1.2 Qualitative research

Looking at qualitative research, it is essentially the reverse of quantitative research. Whereas quantitative research uses numbers and statistics, qualitative research mainly focuses on words and phrasings in order to draw conclusions (Hancock et al., 2021, p. 5). Qualitative research fills in the hole left by quantitative research, where the variables that previously needed to be restrained can now be the primary target of the investigation. "It is these qualitative areas in social life - the backgrounds, interests and broader social perceptions that defy quantitative research - that qualitative research addresses." (Holliday, 2007, p. 5). Holliday then goes on to promote the aspect of discovery through qualitative research, saying it encourages open-ended questions that may lead the researcher to further questions and answers they themselves did not foresee. This ties in to what Hancock et al. (2021, p. 7-8) explained, in that if there is little to no previous knowledge of the topic being researched, then qualitative research is more suitable than quantitative research. They also explain that qualitative research is suitable for situations where its important that the data

collected comes from the perspectives of the participants, and when words rather than numbers are to be analysed.

3.1.3 Qualitative vs Quantitative

Taking all the aspects presented into account, the choice was clear. Note that no option is wrong, and that both research methods have their own strengths and weaknesses, but in our case, qualitative research was the obvious answer. There hasn't been any previous research done on the topic, meaning there aren't any variables that has already been identified as significant, which would make quantitative research difficult to conduct. This study also aims to capture the students knowledge and understanding of programming in such a way that numbers and statistics would not suffice. It was important to collect the data straight from the students and give them the opportunity to reflect on their understanding in ways they themselves may not have thought before.

Of course, there is nothing that is preventing us from using a mix of both qualitative and quantitative research in this project, however, as mentioned previously, the problem would be that the topic of research is relatively new. With no clear variables to analyse, the data obtained through a survey may not end up fruitful and a lot of time would be spent on little to no results. Therefore, this project focuses solely on qualitative research.

3.2 Case study

Having chosen qualitative research as the research methodology, the next step would be to determine what approach to the research the project would take. There are a number of different ways of executing qualitative research, for example one could perform a biographical study where a person and their life is the focus, or a ethnographic study where the focus is on a culture or social group (Hancock et al., 2021, p. 9). This shows that the approach is dependant on what and who the research subjects are, and in our case the most fitting approach is a case study.

3.2.1 What is a case study?

As Merriam et al. (2002) proclaims "the unit of analysis, not the topic of investigation, characterizes a case study." When conducting case studies, the focus of the studies are on "bounded systems", often groups of people with some shared characteristics (Hancock et al., 2021, p. 9). Here, the researcher's

objective is to analyse and understand some phenomenon surrounding the target of the study. This method has been used in a lot of different fields to further research, such as psychology, education, political science and social work to name a few (Yin, 2009, p. 4). Hancock et al. (2021, p. 15-16) introduces a few other characteristics signifying a case study, one of which being that the research in a case study is performed in its natural context. For example, a study about the happiness of the inhabitants of a local town would naturally be conducted in that town using the people living there. Other characteristics mentioned are case studies often using a mix of different information sources, as well as case studies being more exploratory than other methods. Yin (2009, p. 9) also notes that case studies are suitable whenever the study aims to explain a phenomenon, when the questions asked are "how"s and "why"s. This shows how case studies can be both exploratory and explanatory in nature, and how qualitative methods have quite a bit of overlap when it comes to their characteristics.

Strategy	Form of research question	Requires control over behavioural events	Focus on contemporary events
Experiment	how, why	Yes	yes
Survey	who, what, where, how many, how much	No	yes
Archival analysis	who, what, where, how many, how much	No	yes/no
History	how, why	No	no
Case study	how, why	No	yes

Figure 3.1: "Conditions for research strategies" Source: COSMOS Corporation as cited by Yin (2009) p. 8

In addition to defining what kind of question the study aims to look at, there are two other conditions brought forth by Yin (2009, p. 8-13) that may impact the choice of method. These conditions are displayed in figure 3.1, and shows how they correlate to different methods of qualitative analysis. Here we can see that "how" and "why" questions may also fit using an experiment or a history as the research method, as these methods all lean more explanatory. In order to differentiate these methods, we have to focus on the other two conditions given.

"Requires control over behavioral event" is the condition separating experiments from case studies. In an experiment, the study would be conducted in an environment where it would be possible to control variables to isolate what variables are relevant to the study (Yin, 2009, p. 11). Case studies, as mentioned before, happen in the natural context of the study, which means the researcher can’t and should not put any restraints on the subjects involved. Its crucial that

the targets of the study gets to behave as they would normally for the sake of not influencing their responses.

Lastly, histories and case studies are separated by the third condition; "Focus on contemporary events". Histories mainly revolve around questions aimed at the past, questions where there are not any individuals left that can report and give statements on the phenomenon being studied (Yin, 2009, p. 11). This would usually mean searching through old documents and artifacts in order to obtain data for the research. When the study focuses on present events with access to people relevant to the research, the case study method would most likely be preferable. Yin (2009, p. 11) further remarks that histories and case studies has significant overlap in that histories could also be done for contemporary events. However, as previously mentioned, case studies adeptly mixes a number of different sources in their analysis, in a way that is unlike typical histories.

Case studies has a lot of overlap with other qualitative research methods, however the points made in this section shows how choosing case study is dependent on the research question at hand. Yin (2009, p. 13) nicely summarizes the situation in which a case study is appropriate below:

- A "how" or "why" question is being asked about
  - a contemporary set of events,
  - over which the investigator has little or no control

3.2.2 What makes our project a case study?

Having declared what amounts to a case study, it would now be feasible to present what in our project makes it a case study. Our research questions asks "How does prior programming experience from science and technology subjects in upper secondary school affect students who attend first year programming courses at university? " which already points us towards case study as a suitable research method. This is a "how" question, aimed at explaining the effects of programming in STEM classes, which fits the first condition noted by Yin (2009, p. 13) in figure 3.1. This is also a broad question which will benefit from having an exploratory research method, which case studies were established to be earlier.

Looking at our unit of analysis, namely first-year computer science students, this also fits with our knowledge of case studies. This is a bounded group of people, each one connected through their choice of studies and through being

the class of 2024. The project is therefore studying an event that is rooted in the present, another condition pointing towards case studies. Of course, as the study were held at the university, the study also fulfills the condition of exploring the phenomenon in its natural context.

Lastly, the question to be answered is impacted by numerous variables that us as the investigators have no control over. Some examples of uncontrollable variables that could impact the results are level of motivation, social factors such as friends and family, and general interest in programming and technology. By using case studies as the research method, these factors will become, instead of an obstacle, a valuable part of the data collection.

3.2.3 Case study design

In order to keep the case study succinct and distinct, the case study uses an embedded single-case design. Figure 3.2 shows the four basic designs for case studies, split between single-case and multiple-case designs, and between holistic and embedded cases (Yin, 2009, p. 46). Our project concerns the context of the updated curriculum, with a focus on first-year computer science education. As the unit of analysis, we have focused on two groups of participants, namely first-year computer science students with and without prior programming experience. This leaves us with a single case, within one context using two units of analysis.

A holistic case design would also be a valid case study design for this project, where both units of analysis are combined into one. However, single-case designs have the vulnerability in that the case could turn into something else entirely during the research period, meaning the data obtained from the units of analysis may not turn fruitful in the end (Yin, 2009, p. 49-50). This threat is minimized by using an embedded design with multiple units of analysis. Although, according to Yin (2009, p. 52), an embedded design could still cause the focus of the study to shift from the overarching research question towards the sub-units of the case study

Using a multiple-case design would also be possible for the study. Yin (2009, p. 53) mentions school innovations such as new curricula as a common use of multiple-case design. As our case only focuses on a single university, it limits the way our findings can be generalized. Having multiple different universities as contexts, more plentiful and varied data would be obtained that could strengthen the conclusion of the thesis. However, this would be significantly more time consuming and difficult for a single investigator to perform, hence why this case design was rejected.

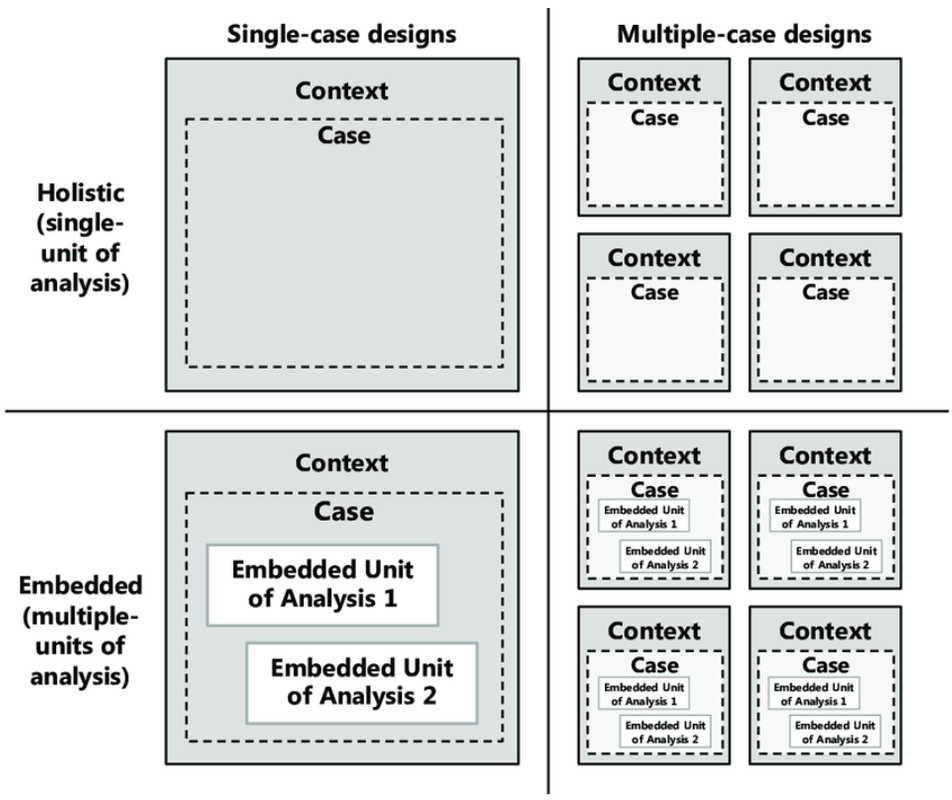

Figure 3.2: "Choice of case study designs "Source: COSMOS Corporation as cited by Yin (2009) p. 46

3.3 Data collection

With both the research methodology and case design chosen for the project, the next step were to decide on how the data for the study would be collected. Both interviews and focus groups were contenders early on, as they both provide direct contact with the participants. However, in order to prevent the subjects from influencing each other, interviews were adopted as the main source of data.

3.3.1 Interviews

Interviews can be viewed as guided conversations, and are often acknowledged as quintessential in case study research (Yin, 2009, p. 106). It provides rich data that should be treated with care and respect due to how personalized it is to each participant (Mason, 2002). There are three forms of interviews to consider, structured, semi-structured and unstructured. Hancock et al. (2021, p. 45) highlights semi-structured interviews as a fitting choice for case studies,

with Mason (2002, p. 62, 67) going as far as to argue that structured and unstructured interviews are unfitting of the title "qualitative interviews". They argue that all research interviews need some structure to follow in order to guarantee content relevant to the research question at hand. On the other hand, having a fully structured plan would result in a methodology more reminiscent of a survey than a case study. By instead opting for a semi-structured approach, the researcher is able to ask predetermined questions that keeps the interview from stagnating, while also enabling the interviewees to talk freely and open up about their own experiences in a relaxed setting (Hancock et al., 2021, p. 45).

With the reasons stated above in mind, a semi-structured interview method was utilized. An interview guide was created with questions aimed at answering the research question at hand. This is an important step in a successful interview, as this provides the interviewer with insightful questions that they can fall back on (Hancock et al., 2021, p. 44). The questions were designed to be open and flexible, such that the conversation is shaped by the interviewees rather than having the interviewees conform to the questions asked. A lot of focus was also put on maintaining informality in the interviews, again to decrease the pressure on the interviewees and let them speak freely.

As mentioned before, the unit of analysis are first-year computer science students at the University of Tromsø (UiT). The only criteria for participating in the interviews were that they were currently studying computer science, this was their first year studying, and they had not studied anything related to computer science previously. Of course, their previous coding experience was also taken into account, as a mix of both experienced and inexperienced students were needed for the case study.

Computer science students were contacted directly by e-mail during the recruitment period. Thanks to our work as a teachers assistant previously, most of the e-mail recipients were students attending the classes, or recommended by those students. In total, there were 25 e-mail recruitments sent to students at UiT. There is a potential drawback, in that some students proclaim that they don't check their e-mails, with some having to be contacted through other means such as text messages.

The emails sent where casual in nature, and quickly presented the project at hand before asking if they were interested in being interviewed. A consent form was also attached to the email, with in-depth explanation of the project and their rights. if they were interested, they were asked to provide a short summary of their coding experience before attending university, as well as a time slot for when they were available for the interviews. The interview was said to last about 30 minutes, a time estimate given in order to obtain a fair

amount of data, while still keeping the interview short and succinct.

Most of the interviews were conducted 12. February 2024, with a later interview conducted 21. March. This is half a year after the students started their studies at the university. This was thought to let the students get a taste of university coding, while still being relatively fresh. The interviews were held at a private room at the university, and were all around 30 minutes long. Before the interviews, a quick recap on their rights as participants were presented. A voice recording was taken during each interview, and the participants consented on tape to the interviews.

Each interview consisted of a conversation following the interview guide made beforehand, as well as a commentary of code snippets provided. There were two code snippets presented seen in figure 3.3 and 3.4, both made in python and both having a simple structure. The students were tasked with explaining the code snippets in whatever way they found fit, without any prompts from the interviewer. If the students were stuck, the interviewer would make note of it, while also asking a guiding question such as "What do you make of this?" or "What is happening in this code?" This section of the interview was done in order to get insight into how the students approach coding, and how they deconstruct and describe written code. The students did not know about any of the questions in the interview guide, and had never seen the code snippets before the interviews in order to keep their responses as natural and genuine as possible.

Figure 3.3: Code snippet 1

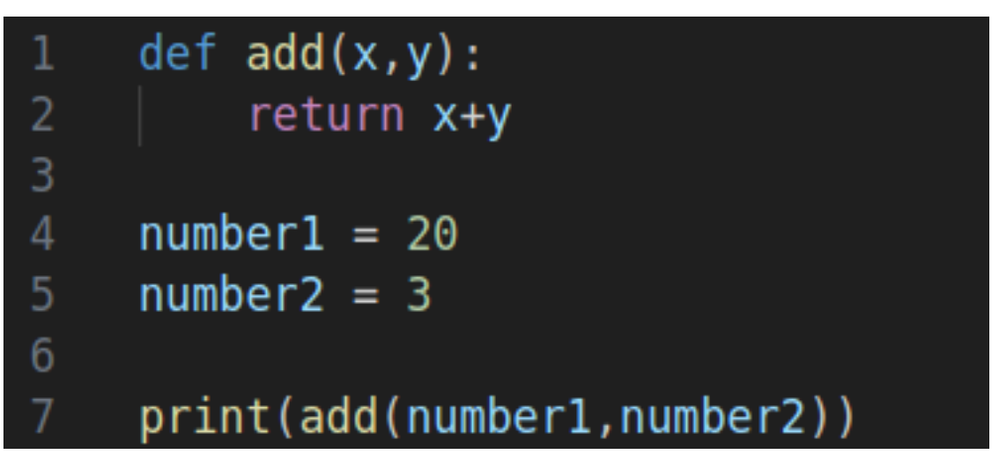

Figure 3.4: Code snippet 2

Having conducted the interviews, a great amount of time was spent transcribing the recordings. The University of Tromsø has an online transcription service named "Klartekst" which was utilized during the transcription. However, the result was not perfectly transcribed, meaning a lot of time was still spent on manually transcribing the parts that were incorrect. These errors may stem from unclear recordings, the varying dialects of the participants, as well as the interviewees and interviewer speaking over each other. The service is also unable to track which participant is talking, meaning this had to be added manually as well.

With both the interviews and transcription done, the data had to be structured and analyzed in a sensible way. Analysing data for a case study is a difficult process, and a number of investigators do not take this into account when beginning a new case study (Yin, 2009, p. 127). With new strategies being developed all the time, some common characteristics has been identified among them. These include the identification of patterns and themes, through vigorous review of the collected data (Hancock et al., 2021, p. 67). Of course, these patterns and themes should then address the research questions that the case study set out to research in some way.

In order to help analyse the data, a qualitative data analysis software was utilized. Yin (2009, p. 127-128) identifies computer-assisted tools as valuable and reliable tools that can help with categorizing and coding the data collected. The analysis is still done by the researcher, however, the outputs gained from the software, such as occurrences of a certain word, may help with noticing patterns in the data. This project made use of Nvivo 14, a complex tool that required a significant amount of time in order to become familiar with. Using Nvivo, a multitude of nodes were created based on the interviews conducted, which helped categorize the answers provided by the students. Some of these nodes were named "Thoughts on programming", "Study methods" and "Prior coding experience". These nodes made it significantly easier to compare responses between the participants.

3.4 Ethics

An important aspect of research involving people is ethics. As stated by Hancock et al. (2021, p. 45), the interviewees should be well informed on their rights, and no deception or damage should take place. As mentioned, the participants received a detailed consent form in which their rights where presented. A short summary of the form was also given before the interviews took place.

3.4.1 Sikt

As the project revolves around people and collects personal data, its important to handle the data in a respectful, safe and legal way. To control this, a form was sent to Sikt - Kunnskapssektorens tjenesteleverandør, an institution that handles research projects concerning personal data. By filling out their notification form with details such as "what data is going to be collected?", "who has access to the data?" and "why is the data collected?", an advisor will review the form and notify whether or not the project follows current data protection legislation requirements. The interview guide and consent form was also attached to the notification form. It was important to get approval from Sikt, as the data collection could not begin without it, therefore a respectable amount of time was spent conveying the research goals and designing the consent form. The form was rejected once, as it did not sufficiently inform the students of their rights during the recruitment period, as well as having some minor errors concerning the date of the project. Having fixed this, the form was approved on the second try.

3.4.2 Anonymity

Every participant in a research project should have their privacy and confidentiality protected. This is to avoid putting them in a troublesome position, such as having their details leaked, or being recruited for other studies (Yin, 2009, p. 73). All names of participants were removed during transcription, and replaced with a identification system using numbers. No data that may identify a person has been depicted in the results of the project. During the study, the data was stored on a cloud service, with access only given to the researcher and advisor. After the end of the project, all data relevant to the study were deleted.

3.5 Validity

With every other aspect of the project planned, there is still the question of "How valid is our data?" In what way can one conclude that the results obtained and the conclusions drawn are reasonable and an accurate measure of what the study set out to research (Hancock et al., 2021, p. 93)?

3.5.1 Generalization

There are two ways of generalizing results, one of which being more relevant to case studies than the other. The less relevant, but more widely known,

is "statistical generalization". This is generalization in which a conclusion is drawn about a population based on a sample of that population (Yin, 2009, p. 38). As this form of generalization has readily available formulas that can calculate how valid the data is based on the size and variety of the population sample, its quite popular in a research setting. However, according to Yin (2009, p. 38-39), this does not fit the setting of a case study. Case studies should use "analytic generalization", where the results should build upon already established theories. It can be thought about in the same way as an experiment. If multiple experiments points towards the same conclusion, then more validity is added to that conclusion. Case studies work the same way according to Yin (2009).

3.5.2 Validity tests

In order to test the quality of the research design created, there are four common tests used in case studies (Yin, 2009, p. 40).

Construct validity

Construct validity tests if the data obtained genuinely reflects what the study aims to research by focusing on the procedures of the study. A concern when doing case studies is that the results may be influenced by the subjective opinion of the investigator, and that the claims brought forward do not portray the phenomenon investigated in a faithful way (Yin, 2009, p. 41). Having multiple sources of evidence, and establishing a chain of evidence are two tactics mentioned by Yin (2009, p. 42) that increases construct validity in a case study. Both of these tactics are present in this project.

Internal validity

Internal validity mainly concerns explanatory case studies, meaning it is of not much relevancy to our exploratory case study. This tests whether the causal relationships concluded by the investigators are in fact correct, or if there is another factor that have caused the results (Yin, 2009, p. 43). Pattern matching and addressing rival explanations are ways of strengthening internal validity.

External validity

External validity tests how generalizable the study findings are, and if its valid outside the boundaries of the case study. As mentioned before, generalizing results of a case study differs from typical survey research, in that the conclusions are generalized unto a broader theory instead of a population (Yin, 2009, p. 43).

Reliability

Reliability concerns how repeatable the study is, meaning that the same result should be obtained if someone where to perform the same investigation using the same procedures. Minimizing errors and subjectivity in the study would increase the reliability. Its therefore important to document the procedures of the case study in detail in such a way that a third party, or the investigator, can repeat the work (Yin, 2009, p. 45).

4

Results

4.1 The selection

After a recruitment process consisting of both meeting students at the common area of the university, as well as a copious amount of emails being sent to first year computer science students, a total number of six students accepted the request to be interviewed. Even though this may be a small sample of people to conduct a analysis, the sample is still diverse when it comes to their experiences with programming.

Of the six students, two of them has had first hand experience with the new curriculum, both having programmed in math class. Of the four other students, two has had absolutely no previous coding experience, one has loosely attended an object-oriented programming class for a single semester which they failed, and one has completed a year long backend online class before attending the university of Tromsø. In order to maintain the most validity with the findings, the latter has mostly been excluded from the collective results. However, some quotations by the student that I have deemed notable for the study will still be provided and discussed.

4.2 Previous experiences

When explaining their experiences with programming, a few similarities in their descriptions presented themselves. The students with previous coding experiences all noted python as the sole programming language they had been exposed to in schools.

4.2.1 Programming in maths

Having coded in maths, two of the students explained how programming was the highlight of their math class. They described programming in the classes as "simple and straightforward" and as a way of doing maths in a digital environment. It was important to note that they did not create big programs, and that they were restricted to performing calculations such as finding the derivative of a function, as well as creating tables and graphs.

Even though both candidates enjoyed and felt mastery working with programming in mathematics, they had a difference of opinion when it came to what they thought of the lectures. One candidate had online classes with a very competent lecturer, and had nothing negative to say about how programming was integrated into the math course. The other candidate however was more critical of their classes, saying they got the sense that the teacher had been to a programming class themselves, and simply relayed the teachings to their own students. Adding unto that, the candidate also felt that programming could have been more prominent in the class, as it mostly ended up being a short summary at the end of each chapter where it was explained how Python could be used to solve the assignments. In addition, some days before the exam, a single day was reserved to exclusively focus on python programming. In their own words, this was described as a "If we've got time"-project.

4.3 What is programming?

A wide variety of answers was given in response to questions aimed at ascertaining their understanding of programming. Candidates without prior coding experience had mostly vague responses layered with some uncertainty. Negative associations with programming was provided by one such candidate, often repeating sentiments such as "it's tough" and "I'm having a hard time". This candidate would also have short answers using singular words to describe programming, such as "money", "video games" and "C", and would also need further probing in order to come up with such answers. A bit more detailed answers where provided by another student without prior experience, though

uncertainty was still present. This student mentioned how they used to think about programming as hacking, however their understanding now is that it's about sorting and handling information. Downplaying their previous answer, they would then proclaim that they don't really know how to define it and have faced issues when friends of theirs have asked to explain their studies. When confronted with the question "what is programming?" the student replied "I can not define it". All candidates with no prior experience also simply said programming is used everywhere with physical examples such as computers, TVs and toasters.

Looking at the responses from candidates with prior experience, the answers were more elaborated and certain. They did not limit themselves to only answering with nouns, and did not hesitate share their thoughts. "Opportunities", "innovation", "language", and "information" are some of the singular words noted. One student elaborated, saying programming gives opportunities to people and makes life easier. They went on to say that this could be helpful in all businesses such as the health sector, education and the military, in which big tasks can be made easier and more approachable using programming. Another student made points backing this, saying programming is a way of "talking" with the computer and expressing what you want the computer to do for you. A more precise area of use was depicted by this student, saying that programming can be used to treat and analyse data, and make data more user-friendly, as well as creating software such as video games and web pages.

One person in the experienced group also made sure to mention that programming can be used in harmful ways, mentioning AI and deepfakes. They noted that these negative ways of handling programming can't be ignored and should be controlled in a way where the least harm is done.

4.4 Associating programming with other subjects

All candidates seemed to connect programming to previous subjects from school in varying degrees. Maths was mentioned by every candidate as a subject that's relevant to programming. Both students with and without prior coding experience noted that there are terminology crossover between the two subjects, such as the derivative, and set operations like union and intersection. The fact that a lot of arithmetic operations is used in sorting algorithms where also mentioned. Even though maths was brought up by all the students, there were still disagreements about how necessary math skills really are in order to do well in the assignments. One student without prior experience was adamant

about how having advanced mathematics in upper secondary schools is a necessity before starting computer science at the university. On the other hand, a student with prior coding experience in math class felt like programming is different and unique enough that they wouldn't really associate it with any subject, even maths. This student was also the only student that answered that they enjoyed maths as a subject. Only one student mentioned a subject other than maths, and said that STEM subjects in general such as physics could be connected to programming.

4.5 Programming at university

Every interview candidate acknowledged that university has been challenging so far, however some differences between the experienced group and the not-experienced group where noticed.

Those with no previous coding experience emphasise how lost they felt when they first started coding at university, especially in relation to the terminology. One such student explained how confused they were when the concept of a terminal was brought up in class, while another described the difficulties of talking with other students and teachers because of their missing vocabulary. Interestingly, the students without prior experience did not feel more lost than other students, and when faced with the question "do you feel at a disadvantage having not programmed before?" everyone simply answered "not really". Another sentiment shared by all no prior experience students were that they wished that the studies where less ruthless at the beginning as this prevented them from getting a deep understanding of the material. One student also made a point of proclaiming C as their favorite language so far, as they found it hard to understand the syntax of python. Its important to note though that the interview was conducted early on in the second semester, which meant that they had not had as much time with Python as they have had with C.

Even though the students with prior coding experiences also talk about the difficulties of their studies, they mostly describe the difficulties of learning a new programming language in the form of C. They find technical aspects, such as pointers and allocation of memory, more difficult while also having a hard time adjusting to working independently. Further elaboration is provided by saying that coding in mathematics is more focused on translating mathematical operations to a code that the computer will understand. Its also easier to know what the end result should look like using python in maths, as you can usually visualize how a graph should look, or what the result of an arithmetic operation is.

Whereas the other group exclaim that they do not have any disadvantages, the group with prior experience is quite clear about them having an advantage. This advantage to them was through their familiarity with syntaxes, as well as importing files and the terminal. One candidate did however note that since there was a huge jump from math class to university, the advantage did quickly fade away.

An interesting remark made by the student with earlier university experience was that they felt like they had an advantage by knowing where to look and what to search for whenever they felt stuck in an assignment. Other students would therefore look to them for answers

4.6 Code snippet observations

No student had problems explaining what happens in the code snippets presented, though there were a few differences in the order they picked apart the code, as well as a varying amount of details in their explanations.

4.6.1 Code snippet 1

During the first code snippet, all candidates with no prior coding experience quickly made note of the coding language being python. They mentioned that they knew that because of the next part they studied, namely the print statement. Since the print statement did not have an "f" at the end of it, they knew it was not C, and therefore must have been python. Lastly, they mentioned the operations inside the print statement and said that the computer would calculate the result and print it to the terminal.

The first thing that caught the eye of the other group was the mathematical operations inside the print statement. They explained that after calculating the result the computer would print the result to the terminal. One student however stumbled a bit being unsure if the result would print without quotation marks, but did conclude that it would. Mentioning the fact that the language was Python was done last by the group.

4.6.2 Code snippet 2

Not much more of interest is added by their commentary of the second code snippet. Every student successfully explained what would happen when the code was ran, and they all immediately started by explaining the defined "Add"

function. One thing of note however is the fact that all people with prior coding experience would start by examining the end of the functions, both what the "add" function returns, as well as what the program prints. Those with no experience would simply have a top-down approach and start at the top of a function and end at the bottom.

4.7 Studying habits

The student's studying habits was also a theme in the interviews, but no clear differences between the groups presented themselves. Both groups had a mix of people that enjoyed working together, and people that would rather work alone, however it did seem that more people without experience would seek teachers and other students to discuss with. The group with prior experience emphasised how they always start by "visualizing" the code and go into the depth of how the code works. This was done by using chatGPT, searching online, trial and error, and creating figures.

Creating figures was also mentioned by the inexperienced group, but was often dropped by them as they felt like it made them lag behind, which would result in them just asking others or the internet for help. Working systematically through the code was something one of the inexperienced students said they were working on, and that they were trying to become better at analysing what the inputs of a function is, and what it outputs. Another inexperienced student mentioned how the group sessions with a teachers assistant were more helpful to them than the lessons, as they learned more through discussing the assignments rather then listening to a description.

4.8 Thoughts on the curriculum renewal

The decision to integrate programming with other subjects in school was met with universal acclaim from the interview candidates. Every student raved about the decision, praising how it will expose more students to programming and make computer science more approachable. This was backed by saying that everyone is constantly surrounded by technology, both privately and at work, and having some knowledge about how programming works is becoming a necessity. No student saw the need for programming to be its own subject, however a few mentioned how they would like it be a offered as a optional subject. One student mentioned how they probably would have pursued programming earlier had it been offered as a subject at their school. Another student made the point of saying that having programming in mathematics

gives a perspective to mathematics that has not been offered before, elaborating that it would help students see the practical use of math.

5

Discussion

This chapter will discuss the findings of the interviews with relation to the research questions. The structure will mostly copy that of the results chapter.

5.1 Integration of programming

A major concern of integrating programming into already established subjects was the fear that it would be neglected because of inexperienced teachers (Sanne et al., 2016, p. 76). This case study observed this first hand as the students with prior experience had varying degrees of satisfaction when it came to their lectures in upper secondary school. Whereas one satisfied student had a competent teacher with previous programming experience, another felt like the programming was dismissed until the last second of the class. However, interestingly, this did not steer them away from programming, and they still considered it their favorite part of the mathematics class. This could point towards teacher competence not having as much effect on the students, as long as they have some ability to teach programming, an observation also made by (Kjällander et al., 2021, p. 12).

Dahl, Ranestad, and Hole (2017) brought forward another concern, namely how programming could hurt the deep learning the curriculum seeks to strengthen. Not only do they explain that adding programming as a new topic in mathe-

matics would further complicate math as a subject and be a disadvantage to pupils already struggling in maths, but they also proclaim that it would change the nature of maths as a subject. Maths aims to explain **why** mathematical methods and operations work, while programming focuses on **how** they work. Our findings may lend credibility to this view, as both candidates with previous experience enjoyed programming in maths class, but one did not enjoy maths as a whole. They also noted that the classes were mostly about translating mathematical operations into code, and since they knew what result to obtain beforehand they spending most of the work on getting that result. As this is not methods typical to math as a subject, this could point towards there being an incompatibility between maths and programming.

5.2 Students perception of programming

When trying to define programming there was a noticeable difference between the two groups. The inexperienced group were much more vague and negative in their responses, often using negatively loaded words to describe their thoughts. Their hesitation and tone also pointed towards them being overwhelmed by the curriculum at university. This is in contrast to the experienced group who provided elaborate and concrete answers to questions pertaining to programming. By using real life examples such as AI, deepfakes and web pages, they managed to connect their understanding to the real world. This compliments a theory mentioned by Kjällander et al. (2021, p. 11), suggesting that earlier exposure to programming positively improves their attitude towards it. The results could also suggest that exposing pupils to programming early on could make them more receptive to seeing the digital landscape that surrounds them.

5.3 Associating programming with other subjects

Interestingly, most students seemed to only associate programming with maths, with no real disparity between the two groups. This could however be greatly affected by their experience at university, as the programming courses they have experienced all makes great use of mathematical operations. As there was only one participant that mentioned programming being unique enough to not be associated with other subjects, this does lend credit to the decision to not make it its own subject in the curriculum. Another interesting fact is that not every candidate agreed that maths was necessary to study computer science,

which would suggest that there is not a strong connection between the two subjects as the curriculum renewal portrays.

5.4 Programming at university

As the study by Bolland (2023a) suggests, we would expect the group with prior experience to have a much easier time at university than the group without experience, however this study does not reflect that. Though the experienced students seem to be a bit more secure in programming as a whole, they experience another difficulty, namely migrating between programming languages. As they were only familiar with python from maths class, they had a hard time switching to C as the programming language. This highlights a concern raised by Sanne et al. (2016), where programming itself isn't learned, but rather the programming language python. This is one of the main factors towards them wanting programming as its own separate subject. As one inexperienced student went as far to say that C was their favorite language, this could suggest that learning programming as a whole before committing to a specific language could be beneficial in order to prevent students from becoming too attached to a single language.

There also seem to be a inconsistency in their understanding of their own programming knowledge, as the inexperienced group felt no disadvantage towards not having programmed before, while the experienced group did feel an advantage over the other students.

5.5 Code snippet observations

The biggest differences between the groups revealed themselves during the code snippet observations. Both the groups had unique approaches to the code snippets, with the experienced group seemingly analysing the snippets with a more algorithmic approach, very reminiscent of the way computational thinking is thought of in the curriculum. They would separate the code into sections, first identifying the functions, then making note of the outputs and inputs of the functions. This was in stark contrast to the other group who simply read the code line by line starting at the top. However, the inexperienced group would be more peculiar at specifying the programming language used, and studying the semantics of the code, such as observing if there were quotation marks signifying a string. This seemingly suggests that a students understanding and processing of code could be influenced by the school subject in which programming is used.

5.6 Studying habits

There does not seem to be any big differences in the studying habits of the two groups. Some of the statements though did seem to suggest that the students with previous experience had an easier time looking for the answer than the other group. Whereas the experienced group would search online and use chatGPT for answers, the inexperienced group would seek out other students and teachers aid in order to get help. This is opposite to a theory mentioned by (Kjällander et al., 2021, p. 13), where students seem to collaborate less in the beginning when working with programming.

5.7 Thoughts on the curriculum renewal

While studies show that teachers have mixed opinions when it comes to the curriculum renewal, all the students in the case study had a positive attitude towards it (Stenlund, 2021). Interestingly, they all specifically said programming does not need to be its own subject, though having it as an elective subject would be nice. Having it in math specifically was also praised as it would help see the practical use of math. This result shows the disparity between teachers and students, and is heavily in favor of the curriculum renewal. However, a lot of the reasoning as to not wanting it as its own subject was because of practical aspects, such as them imagining it to be hard to apply a new subject to schools in this day and age.

5.8 Weaknesses of the study

As this is a case study working with humans of all kinds, there are a few weaknesses in the study that may have affected the results. This section will identify those weaknesses.

Firstly, there are no participants of the study that have experienced programming for more than a year before starting university. In order for students to have experienced the curriculum renewal in its entirety, they would have to have graduated from upper secondary school in 2023. There was no guarantee that there would be any students who fit this criteria currently enrolled in first year computer science class, and there was further no guarantee that they would accept taking part in this study. There is a possibility that this has influenced our result and that having data from students who graduated in 2023 would have created a bigger divide between the two groups interviewed.

As the students were chosen based on their willingness to participate, some viewpoints may have gone unrepresented in the results. For example could there be students not confident enough in their programming knowledge that declined the invitation, causing our results to skew towards those with more understanding. A lack of confidence could also play part in the actual results obtained, as there may have been some students altering their answers in fear of saying something they perceived as wrong.

In this project, the data collected was limited to only one university. This could affect how generalizable the result is, as the level of competence may vary between universities. Another factor to consider is that the students all come from different parts of Norway, meaning

Looking at the participants, there are a number of outside factors that could have influenced the results rather than their previous programming experience. Fiddling with programming and technology at their own accord could be such a factor. Some of the participants made note of their interest in gaming, with one of them mentioning their experience setting up game servers. Experimenting with hacking activities were also mentioned. Other factors that could play part in the results obtained could be them being exposed to programming through friends or family, or their academic proficiency.

As this project is about the curriculum renewal, a fairly new event, it has been difficult gathering data from earlier research to complement the results obtained. Though some research has been identified and discussed in this assignment, there may still exist studies not mentioned in this project. This is also the case for studies from other countries.

This is the first time the researcher has performed interviews, which could have affected the quality of the data obtained. Conducting interviews are a skill that can be refined over time, where a couple of questions could be the difference between a successful and inferior interview. This could possibly have resulted in interesting trails of questioning not being followed up, or leading questions being asked. However, by creating and following a interview structure, these biases has been reduced as much as possible.

A lot of time was spent on getting our notification form approved. This had to be approved before any data collection were to be done. The interview guide and consent form had to be constructed, and all details of the study had to be filled out and sent to Sikt in order to obtain the approval. As the form was rejected once, even more time had to spent on reviewing and changing the form before it finally got approved.

5.9 Validity

As validity is an important aspect of a successful case study, this section will describe what measures was taken in order to maximise the degree of validity:

5.9.1 Construct validity

Two different data sources was used in this project. Both data collected through semi-structured interviews, and earlier experiences in Norway and other countries was used in order to conduct the case study.

5.9.2 Internal validity

As this is an exploratory case study, internal validity is not of much relevancy. However, in order to keep the two units of analysis comparable, the groups were limited to first year students currently enrolled in computer science without earlier university-level programming education.

5.9.3 External validity

Our findings can contribute to existing research and theories ascertained to programming in education. The theories that built up this study could potentially be analytically generalizable if our findings are complimentary. The research done in this project only represents first-year computer science students at university, with a focus on those who have and have not been affected by the changes introduced by the curriculum renewal.

5.9.4 Reliability

As this case study were of a smaller scale, a case study protocol was not created, however the steps followed during the project has been documented in chapter 3. In the event that someone were to replicate the study, they would make special note of chapter 3.3.1, as this is where the steps for the interviews are described. All data collected is safely stored and could be expected by an external unit.

/6

Conclusion

This case study has explored the application of a new curricula in Norwegian schools, with a focus on two research questions involving students' relation to this change. Data has been collected through a brief review of the renewal itself, an exploration of how programming has been applied to schools in other countries, as well semi-structured interviews of six first year computer science students.

Firstly, the study aimed to answer how the curriculum renewal has affected students who now have experienced programming through mandatory subjects at upper secondary school. In order to create a comparison, both students with and without prior programming experience was interviewed, with two students having experienced one year of programming in mathematics, and four students having no prior experience from upper secondary school.

Students with prior experience talked highly of the programming they learned in math class, in contrast to how they perceived the competence of their teachers. There was a varying degree in how competent the teachers was, ranging from highly skilled to simply relaying a programming class they themselves attended. Though they enjoyed programming in math, they did not necessarily enjoy math itself, and described their work with coding in math as simply translating mathematical operations into python. This could point towards a lack of connectivity between math and programming, and does not reflect the motivation behind the curriculum renewal, namely increasing deep learning in math.

An increased confidence was observed in those with a programming background in math. Having prior experience meant they were able to confidently describe programming, using both technical speech as well as connecting it to real world examples such as AI, deepfakes and web pages. Inexperienced students would hesitate defining programming, and struggled connecting it to the real world.

Though the experienced group felt they had an advantage over other students, this was not necessarily observed in the study. Having only programmed in python, the experienced students faced difficulties having to migrate to a new programming language in the form of C, and felt that most of their advantage came from knowing python. The inexperienced group did not feel any disadvantage having not experienced programming before, and noted that it quickly evened out.

When faced with code, the experienced group approached it with a more algorithmic approach than the inexperienced group. The outputs and inputs of a function were studied before the function itself was observed. This is reminiscent of what the curriculum renewal refers to as "computational thinking". The inexperienced group on the other hand, would start by determining what programming language was used, before reading the code line by line, starting at the top.

Students with experience seem to not seek as much help from others as the inexperienced group. This could be a result of them finding it easier to search up the answer online as they know what they should look for. In contrast, inexperienced students would make more use of teachers aides and fellow classmates in order to obtain help. No other differences in their studying habits were noticed.

The study also managed to get a view into how the students themselves feel about integrating programming into already established subjects, with them all having a positive attitude towards it. All students explained that learning programming is a necessity for the future, and that adding it to mandatory classes would make sure all pupils would get exposed to it. Some also argued that adding it to math class would help show how maths can be used practically.

The integration of programming in Norwegian schools had the goal of introducing deep learning and computational thinking to students, which this study observes mixed results of. Students with prior programming experience seem to be more secure in their knowledge of programming, and reads code more systematically, however, they are closely tied to python as a programming language, and therefore found university difficult when other programming

languages were brought up. They are also notably more positive towards programming as a whole, though all the students where positive to the introduction of programming in established subjects. This study finds that though there may be worry caused by lacking programming competence from teachers, this does not affect the students' motivation. Hopefully, this will result in future generations that are capable of keeping up with the digitization of society.

Future research

As of the time of this study, the curriculum renewal has taken full effect and every grade from elementary school to upper secondary school has been impacted. This means there are now multiple grades that gets to experience programming in Norwegian schools. It would be interesting to perform the same study on pupils at elementary school in order to see the effects of programming at lower grades. Performing the study on non-computer science students would also be interesting as this could give insight into how programming is understood by students not pursuing the field.

References

Armstrong, P. (2010). *Bloom's taxonomy.* Vanderbilt University Center for Teaching. Retrieved from https://cft.vanderbilt.edu/guides-sub-pages/blooms-taxonomy/ (Accessed: May 31, 2024)

Bocconi, S., Chioccariello, A., & Earp, J. (2018). *The nordic approach to introducing computational thinking and programming in compulsory education* (Tech. Rep.). Report prepared for the Nordic@BETT2018 Steering Group. Retrieved from https://doi.org/10.17471/54007

Bolland, S. (2023a). National prior knowledge test in programming-how proficient are incoming higher education students? In *Norsk ikt-konferanse for forskning og utdanning.*

Bolland, S. (2023b). *National prior knowledge test in programming-how proficient are incoming higher education students?* Retrieved from https://programmeringstesten.no/

Borge, I. C., et al. (2014). *Matematikk i norsk skole anno 2014 - faggjennomgang av matematikkfagene - rapport fra ekstern arbeidsgruppe oppnevnt av utdanningsdirektoratet.* Retrieved from https://www.udir.no/globalassets/filer/tall-og-forskning/rapporter/2014/matematikk_norsk_skole_2014_rapport_ekstern_arbeidsgruppe.pdf (Accessed: March 29, 2024)

Dahl, G., Ranestad, K., & Hole, A. (2017). Programmering rammer dybdelæring i matematikk. Retrieved from https://www.aftenposten.no/meninger/kronikk/i/E0zga/programmering-rammer-dybdelaering-i-matematikk-geir-dahl-kristian-ranestad-og-arne-hole (Accessed: May 31, 2024)

Department for education. (2013). *National curriculum in england: computing programmes of study.* Retrieved from https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study (Accessed: May 03, 2024)

Hancock, D. R., Algozzine, B., & Lim, J. H. (2021). Doing case study research: A practical guide for beginning researchers.

Holliday, A. (2007). Doing & writing qualitative research.

Kjällander, S., Mannila, L., Åkerfeldt, A., & Heintz, F. (2021). Elementary students' first approach to computational thinking and programming. *Education Sciences*, *11*(2), 80.

Kunnskapsdepartementet. (2016). *Meld. st. 28 fag – fordypning – forståelse — en fornyelse av kunnskapsløftet til innholdsfortegnelse kun.* Retrieved from https://www.regjeringen.no/no/dokumenter/meld.-st.-28-20152016/id2483955/ (Accessed: March 30, 2024)

Kunnskapsdepartementet. (2017). *Framtid, fornyelse og digitalisering - digitaliseringsstrategi for grunnopplæringen 2017–2021.* Re-

trieved from https://www.regjeringen.no/no/dokumenter/framtid-fornyelse-og-digitalisering/id2568347/ (Accessed: March 30, 2024)

Kunnskapsdepartementet. (2019a). Læreplan i kunst og håndverk (khv01-02). (Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2020)

Kunnskapsdepartementet. (2019b). Læreplan i matematikk 1–10 (mat01-05). (Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2020)

Kunnskapsdepartementet. (2019c). Læreplan i musikk (mus01-02). (Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2020)

Kunnskapsdepartementet. (2019d). Læreplan i naturfag (nat01-04). (Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2020)

Kusaka, S. (2021). Systematizing ict education curriculum for developing computational thinking: Case studies of curricula in the united states, australia, and the united kingdom. *Journal of Education and Learning*, *10*(5), 76. Retrieved from https://doi.org/10.5539/jel.v10n5p76

Mason, J. (2002). *Qualitative researching* (Vol. 2). Thousand Oaks, CA: Sage.

Merriam, S. B., et al. (2002). Introduction to qualitative research. *Qualitative research in practice: Examples for discussion and analysis*, *1*(1), 1–17.

NOU 2013:2. (2013). *Hindre for digital verdiskaping.* Retrieved from https://www.regjeringen.no/no/dokumenter/nou-2013-2/id711002/ (Accessed: March 29, 2024)

NOU 2015:15. (2015). *Digital sårbarhet – sikkert samfunn — beskytte enkeltmennesker og samfunn i en digitalisert verden.* Retrieved from https://www.regjeringen.no/no/dokumenter/nou-2015-13/id2464370/ (Accessed: March 30, 2024)

NOU 2015:8. (2015). *Fremtidens skole — fornyelse av fag og kompetanser.* Retrieved from https://www.regjeringen.no/no/dokumenter/nou-2015-8/id2417001/ (Accessed: March 29, 2024)

Regjeringen. (2017). *Overordnet del - verdier og prinsipper for grunnopplæringen.* Retrieved from https://www.regjeringen.no/contentassets/53d21ea2bc3a4202b86b83cfe82da93e/overordnet-del---verdier-og-prinsipper-for-grunnopplaringen.pdf (Last updated: September 07, 2017, Accessed February 15, 2024)

Sanne, A., et al. (2016). *Teknologi og programmering for alle.* Retrieved from https://www.udir.no/globalassets/filer/tall-og-forskning/forskningsrapporter/teknologi-og-programmering-for-alle.pdf (Accessed: March 30, 2024)

Sentance, S., & Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and information technologies*, *22*, 469–495.

Skolverket. (2018). *Curriculum for the compulsory school, preschool class and school-age educare.* Retrieved from https://www.skolverket.se/download/18.31c292d516e7445866a218f/1576654682907/pdf3984.pdf (Accessed: May 02, 2024)

Statped. (2021). Programmering. Retrieved from https://www.statped.no/laringsressurser/teknologitema/programmering-for-barn-med-saerskilte-behov/programmering/programmering-i-skolen/ (Last updated: March 1, 2021, Accessed March 13, 2024)

Stenlund, E. (2021). *Programmering og fagfornyelsen* (Unpublished master's thesis).

Uddannelses- og Forskningsministeriet. (2023). *Danmarks digitaliserings strategi - ansvar for den digitale udvikling.* Retrieved from https://www.digmin.dk/Media/638357207253210400/SVM%20regeringen_Danmarks%20digitaliseringsstrategi_2023_V9_Online_Final%20(1)-a.pdf

Utdanningsdirektoratet. (2019a). *Algoritmisk tenkning.* Retrieved from https://www.udir.no/kvalitet-og-kompetanse/digitalisering/algoritmisk-tenkning/ (Last updated: March 27, 2019, Accessed February 15, 2024)

Utdanningsdirektoratet. (2019b). *Dybdelæring.* Retrieved from https://www.udir.no/laring-og-trivsel/dybdelaring/ (Last updated: March 13, 2019, Accessed February 15, 2024)

Utdanningsdirektoratet. (2021a). *Kunnskapsløftet 2020 – hvorfor har vi fått nye læreplaner?* Retrieved from https://www.udir.no/laring-og-trivsel/lareplanverket/fagfornyelsen/hvorfor-nye-lareplaner/ (Last updated: June 24, 2021, Accessed February 15, 2024)

Utdanningsdirektoratet. (2021b). *Slik ble læreplanene utviklet.* Retrieved from https://www.udir.no/laring-og-trivsel/lareplanverket/fagfornyelsen/slik-ble-lareplanene-utviklet/ (Last updated: September 22, 2021, Accessed February 15, 2024)

Vogt, Y. (2021). Programmering blir allemannseie i skolen. Retrieved from https://www.apollon.uio.no/artikler/2021/1_utdanning_programmering.html (Last updated: February 1, 2021, Accessed February 17, 2024)

Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). sage.