NTNU
Norwegian University of
Science and Technology

Thematic exercises for non-IT students in Introductory Computer Science courses

Andreas Haugan Aursand

June 2019

Supervisor: Professor Guttorm Sindre

Master's thesis
Department of Computer Science
Norwegian University of Science and Technology

Til morfar

Summary

With the rapid increase in digital technology over the last few decades, learning programming skills for application in a broad range of scientific fields have never been more important. Students at The Norwegian University of Science and Technology (NTNU) are given fundamental knowledge of computer science through the introductory course: Introduction to computer science (TDT4110). And while the course itself covers a vast amount of basic knowledge, there is uncertainty towards students with non-IT majors on if they are fully aware of the number of applications that programming can be used in the students primary field of study. This thesis investigates the possibilities of implementing thematic exercises for non-IT students, to better demonstrate the applications programming can be applied to inside the scientific area. The research is conducted as a survey in the form of an online questionnaire and a series of interviews with students. The findings show that students show positive attitudes towards the implementation of exercises, but the theorized solutions are complex and needs substantial time and resources to implement. As of the writing of this paper, students are more interested in including other aspects of programming that they find useful, one of these aspects being graphs and plotting in Python

Preface

I would like to thank my supervisor, Guttorm for invaluable advise and help during many months of frustration, intense work hours and writer's block. I would thank everyone that participated in the questionnaire and subsequent interviews and for sharing their opinions. I would also like to thank the other master students in my study hall for discussions, laughs, breaks and social activities to take our minds of the thesis when it was needed. To anyone from the future that discovers this thesis: Writing a Master's thesis can seem daunting, scary and even downright hellish at times. My best advice would therefore be to sometimes take a step back and take a few breaths, just a few, before continuing on. Even a breather for 10 seconds can turn around a tiered mind in late evening hours, and give oneself renewed vigor to continue on.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Modern society is more connected than ever before, with the expanding and continually evolving technological area of software technology. In the same fashion, the need for software developers has increased in the last few years. ICT Norway (IKT Norge) has predicted that there are over 6000 vacant positions today [25]. Also, the Directorate-General for Employment, Social Affairs, and Inclusion, a part of the EU commission has reported substantial shortages of Software developers and systems analysts. In a report from 2017 [29], there were findings of significant work shortage throughout the EU-sphere, as well as other countries with ties to the EU, such as Norway, Switzerland, and Iceland.

Since the turn of the century, technology has exponentially increased in number and variation, both hardware and software areas, as well as expanding into other areas of society. With this rise, there is also a clear need for the new generation of students to have a vaster knowledge of Computer science (CS) and how this can relate to their main disciplines. The increase has also been a result of the industry itself, having to turn to other study programs to fetch engineers with some CS skills that they can utilize.

Also, the introductory course at the Norwegian University of Science and Technology (NTNU) has become a mandatory course for most students, where the first-year students participate in the class, regardless of the study course. This means that students from biology, physics, and Computer science all take the same course, even though they might apply the knowledge from this course in different ways. This also means that there are large numbers of students taking the class, and will inevitably lead a limited number of practice tasks that may not correctly demonstrate the possible applications that this course can give most students. This thesis will research the possibilities for introducing the practice exercises for Introduction to Computer Science (TDT4110) to other course themes, and what

technological expansions can be brought to the course that can affect student performance.

1.1 Background

Introduction to Computer Science (TDT4110) is an essential course that every science and technology student at NTNU takes in their first semester. The course covers the basic understanding of how programming works, and how to create programs to solve simple programming exercises with process-oriented programming [4]. Both science and technology students have a mandatory to take the course, and while not all study programs appear to have the same use for the knowledge, it is still stated as a part of these programs. A Chemistry student might not have the same use for programming knowledge with their other courses as a computer engineering student but are still stated as a fundamental part of their study program [23].

So why would one have both chemistry students and computer engineering students have the same mandatory course when only one has an apparent visible need for the course? The main explanation appears more evident when one delves deeper into the other classes of these science courses. A chemistry and physics student has several complex mathematical and other abstract calculations to solve problems. Solving a complex physics problem that might require simulations is possible when one has access to a computer that can do this. It is in essence, a support course for students, so that they may be more robustly prepared in later years of their studies or after university.

At NTNU, one also has a standard set of courses one has to include in the study programs of all Master of Science degrees or a Bachelor degree. By Norwegian law [5], all engineer degrees in Norwegian education has to abide with specific learning outcomes an engineering student will need to master to graduate. The regulation includes that specific knowledge of digital systems and computer competence is to be included in the study program.

Also, by having a broad fundamental understanding of different science topics, there is a much easier way for students to switch degrees if the situation arises [6]. This saves both the student and the university time and money by making the student qualified for entrance into other study programs, should they change their mind.

There also exist several different types of tutorials on the internet that is tailored to teach the basic understandings of a plethora of different topics. These tutorials, like Codecademy [11], Udacity [17], Khan Academy [10] and Coursera [3] all exists to teach either Physics, Chemistry, Programming or other types of computer science, are often widely used to demonstrate, and some times even give

students practice exercises to complete. They do not, however, give sufficient substitute on why one would learn a programming topic concerning a whole university course.

1.2 Motivation

In this section of the paper, the researcher will present the motivation for delving into the automation of exercises for a first-year course and why it may help students improve their knowledge and understanding of computer science. The researcher will also present research questions that The researcher will seek to answer.

First of all, the current course at NTNU, Introduction to Computer Science (TDT4110), does an excellent job at delivering engaging practice exercises. Throughout the course, students must deliver mandatory practices for them to be able to take the exam at the end of the semester. The typical requirements are to complete 8 out of 10 exercises, where each exercise covers a chapter or fundamental theme of computer science and programming. The exercises themselves are divided into separate single tasks with a varied difficulty tailored to give different challenges.

While these exercises alone can give an excellent introduction to computer programming, there is still a limited amount of exercise that the students can use for practice. Also, there is a lack of exercises that demonstrates different applications that one can use computer programming in connection to other fields than computer science. There would be apparent usefulness to teach math or chemistry students how they can use these skills in some way that can benefit them.

Figure 1.1 shows how one topic is presented for the student, where the topic for each exercise listed in the second column, with a separator on the difficulty in the third column.

Oppgave	Tema	Vanskelighetsgrad
Teori - Digital representasjon	Teori	
Grunnleggende om funksjoner	Funksjoner	
Varierte funksjoner	Funksjoner	
Lokale variabler	Funksjoner	
Globale variabler	Funksjoner	
FlowerPower	Funksjoner, Løkker, Grafikk	
Euklids algoritme	Funksjoner, Algoritmer	
Primtall	Funksjoner, Løkker	
Multiplikasjon	Funksjoner, Løkker	
Den store spørreundersøkelsen	Funksjoner, Løkker	⭐
Arbeidsdager	Funksjoner, Løkker	⭐
Sekantmetoden	Funksjoner, Løkker	⭐
Not quite Blackjack	Funksjoner, Løkker, Betingelser	⭐

Figure 1.1: List of tasks from an exercise in TDT4110, including both theoretical and practical.

Based on these motivations, this masters thesis goal is to survey and research the possibilities for expanding the exercises that exist in TDT4110 course to contain other thematic exercises for non-IT-students at NTNU. These would act as motivators for students to learn more and get better results in the course. The research questions are:

- How can thematic exercises help improve students' performance in introductory programming courses?
- Can these tasks be, in some way, implemented into the existing system, and if so, can they replace the current exercises?

1.3 Thesis structure

Following this chapter, The thesis will divide the chapters into the following:

- Chapter 2: Background literature. This chapter will discuss and highlight studies done about the topic, as well as some similar online applications that exist today.
- Chapter 3: Basic Theory. This chapter will include methods used, pros and cons, as well as ethical reasoning and additional theory definitions related to the paper.
- Chapter 4: Survey. Here, the paper will present and explain the details for the data gathering methods, as well as some initial results.
- Chapter 5: Analysis and Discussion. The chapter will consist of analysis and discussion of the findings from the data gathering methods presented in chapter 3 and 4.
- Chapter 6: The final chapter will conclude, and summary findings, as well as limitations and further research, applied concerning the thesis.

Chapter 2

Literature Review

This section of the thesis will comment and review past papers on the area of exercise automation and thematic programming exercises, to get a better understanding of how this area of programming and learning have been researched and developed over the last two decades. In particular, the thesis would want to focus on papers or theses that have delved into the part of automation and thematic exercises. There would be preferable to have information on how other researchers have tried to answer the questions in the past. There will first be a small introduction to the concept of systematic literature review and general in 2.1, before introducing and explaining each literature in 2.2

2.1 Introduction

Systematic literature reviews are a means of identifying, evaluating, and interpreting all available research relevant to a particular research question or topic area. This description is the definition given by [27]. A systematic literature review is a form of secondary study that one performs for several reasons.

The first and foremost reason is to summarize the existing evidence concerning the research topic. In other words, one summarizes the empirical evidence of benefits and limitations on a specific topic. The literature review also acts as a framework/background that the thesis or paper uses to build its theories and research activities. The literature review can also be used to examine the extent to which empirical evidence supports or contradicts the hypothesis of the thesis. The literature review is, in other words, a starting line for the research paper.

Further down in this section, there will be a presentation of motivation for using parts for doing a literature review, before explaining how the search was conducted.

2.1.1 Motivation

The motivation for performing a literature review is to primary research methods and experiments that have been performed in the past, both recent and more distant. This review will not be an accurate systematic review in the sense of time constraints and effort that is needed to be put into the research and analysis of a large number of papers. There is, however, motivation for doing this.

The main reason is to get an overview of how universities teach programming and programming methods to students. Have they had any problems with teaching non-major students in computer science? Do they offer courses to non-major students at all? Moreover, have they taken steps to improve their teaching methods to students? There is also the possibility that some universities have already implemented a version of programming for non-majors. It would be interesting to see how they have implemented the said course and how the structure is applied.

2.1.2 Goals

To better define the literature review, there is a clear benefit to having a set of goals defined, so to not get lost in constant searches of papers that might not have any clear relevance towards the RQs. Since the goal of this paper is to research the potential benefits of adapting programming exercises to non-major students, the goals should focus on narrowing down the search field in which to gather background literature.

- Review goal 1 (RG1): Create an overview of the current situation in educational IT programs and how researchers have iterated on their teachings, either through methods or technological solutions.
- Review goal 2 (RG2): Look for any related research that has focused on non-major students or similar focus groups, and see what solution has been tested and failed.

2.1.3 Literature search

Literature searches are the center point of literature reviews. Finding correct and related information to the thesis is the primary way to get a clear overview of the current situation of the topic. In this day and age, there exist several search engines that contain a vast library of research. The big problem lies in finding

correct material to the topic. Since this is an ad-hock version of a systematic literature review, the following subsections will describe the following origins for the literature summaries that will be discussed later in this chapter.

Searched libraries

The online libraries used was primarily Google scholar and IEEExplorer. It is important to note that there is technically only one database here, IEEExplorer, that is an actual research paper search engine/database. Google scholar does not host any papers directly, but indexes other sources, based on the search criteria. Often, the results that came from Google scholar was both more relevant and had a larger result than using the search engine of IEEExplorer. Since Google scholar was the main tool that the researcher relied on, there were also several other research databases involved.

Library	URL
Google Scholar	https://scholar.google.no/
IEEExplore	https://ieeexplore.ieee.org/Xplore/home.jsp
ACM digital library	https://dl.acm.org/dl.cfm
Oira NTNU	https://www.ntnu.no/ub

Table 2.1: List of databases/search engines

Search terms

Search terms were of utmost importance in finding the relevant texts and papers. In the beginning, there was no direct attempt to create a search matrix to pinpoint a more filtered search. The most used term was ”programming tasks,” that quickly were revealed to give misinforming results, since the term ”task” is considered as a term within AI training, and the correct term that should have been used was ”exercise.” After this mishap was corrected, there was a noticeable change in the search results, and more relevant text started to return from the search query. This term, along with ”Information technology,” ”Education,” and ”programming exercise” yielded better results.

The standard search term used at the start of the project was the following:

> *(((((Computer science) AND exercise) AND programming) OR non-major) AND student)*

These terms usually resulted in some relevant results, but mostly was too vague to discern any consistent results to use in the thesis. After some time, there was

an attempt to directly search through several databases using parts of the original search terms. There was also some help from the thesis supervisor to suggest other relevant literature.

2.2 Reviews

This section will present and review six different papers and relevant sources and discuss their relevance to the thesis.

2.2.1 QuizPack

First, there have been several research papers and theses that have looked into automating exercises for new Computer science (CS) students. Brusilovsky and Sosonovsky 2005 [20] Have developed and tested an automated exercise system that focuses on presenting students with programming problems in the C language. The system, QuizPACK (Quizzes for Parameterized Assessment of C Knowledge) takes parameterized questions as a template, populates the parameters with random values, and presents this to the students as a simple programming exercise they have to solve. The student would then input the answer, and the machine would then compare it to the solution.

The main preface presented in the article was the problem the paper authors observed in the intro course for computer science at the University of Pittsburgh. Since there is a considerable effort to maintain quizzes, as well as small practice exercises over time, there is often a limited supply pool of questions and varied types of exercises that the faculty can maintain. This again results in the quizzes being short and repetitive and also invites cheating. Having a system in place that can instantly generate a new question for students as they take quizzes would hinder this practice, and also lessen the amount of work applied into the quiz-pool for professors and other staff.

The quiz-system, QuizPACK, was in testing over several years at the University of Pittsburgh, and in that time, had several revisions, including speculation of developing a version for hand-held systems, that was later dropped because of technical problems. Overall, the system saw very positive opinions for using it out-of-class as a self-assessment system, that would be available as an additional resource that students could make use of in their own time. The system was poorly received as a replacement for the existing exercises used by the university at the time. This shows that a possible solution for extra exercises would be better received from students.

2.2.2 CloudCoder

Other researchers have also tried similar approaches to the QuizPack system. Papancea, Spacco, and Hovenmeyer 2013 [32] did a paper that describes an open platform for solving short programming exercises. CloudCoder would be an open-source system where users could create, solve, and share programming exercises in a variety of different languages. One of the major motivations for this was to study how students learn programming skills. [32, p.47] This was partly done throughout the CloudCoder's edit history function that kept backlogs of students editing their code in the online browser editor. For the testing and evaluation purposes of the system, the solution presented was to write tests that would partially pass if students did have some correct answers when evaluating the exercises. It was also proposed that there would be some form of "crowd-sourcing" the tasks by having several independent instructors contribute several different exercises to a pool, and ultimately growing the number of different exercises that students could pick and choose.

The exciting solution here is the choice of using crowd contribution and open-source philosophies to expand and improve upon the already existing choices for student exercises. This solution is, among other sites, done by existing sites on the web like Codewars [1], that expands its library of exercises with the option of having its users to contribute with code, exercise texts, and test cases.

2.2.3 Computational thinking

People have also used intro courses in computer programming for more general uses. One would wonder if a different educational approach to exercises would yield different results. Yeh, Xie, and Ke [34] have researched using computer programming courses to teach non-majors computational thinking. They argue that computational thinking (CT) has become a prominent problem-solving method in the past decades, and has shown a more prominent featuring among people taking academic courses, such as maths, physics, and chemistry. They argue that computer literacy today has grown beyond and expanded outside of the normal area. Many jobs of today require computer skills to a certain degree or CT as a method of work that can help in the work environment:

> Modern technologies are so pervasive that computer literacy is no longer merely for a small number of computer and information technology professionals. Many jobs either require computing skills or benefit from CT in today's society. CT can help learners of all dis-

ciplines develop analytical skills and problem abstractions that help them solve their daily problems on the job. [34, p.1]

Yeh, Xie, and Ke explain that the essential skill for CT, abstraction is pivotal to learn the rest of the discipline. They used a more general method of testing learning skill in that they used Microsoft Excel and its functions to teach students CT. They conclude their research and tells that the teaching of such skills is challenging for educators and teachers, and said that there would need to be considerable effort if the CT mindset should be implemented as a core concept in higher education. The exciting approach here was that the researchers did use a different approach to teach core concepts to non-IT-major students, utilizing a somewhat familiar program that many are familiar with Excel, and then applying a completely different method to solve problems and learn core IT-concepts. Moreover, while the conclusion was rather non-conclusive, there are certainly some points that would be interesting to investigate further.

2.2.4 Interactive Technology

Mitra et al. Has written a paper about testing out interactive technology on non-major university students taking introductory courses. Because of the rise in demand for programming skills in nearly all disciplines [30], they have catered to these demands by offering several programming courses that allow students to be certified after completing four of them. The main difficulty they discovered was the large drop-off because of the great intellectual exercise that the programming paradigms are, and have even noticed students with prior knowledge having some difficulty.

The solution for Mitra et al. was to make use of ”Active Learning” along with a Classroom Performance System (CPS) to see if this would increase the students learning as well as enjoyment with the course, to raise exam results and decrease drop-off rates. ”Active Learning” is a technique that forces students to take part and engage in classroom or lecture activities actively. Learning takes place when the students make an effort to process and digest new material. Connecting the information to previous knowledge, summarizing and putting the ideas in the students own words are some of the examples that [30] gives as a clear advantage with Active Learning. CPS is an interactive response system where students, using remote control devices, gives feedback on the lecture, answers quizzes, and gives performance ratings.

The first thing to note is that the article is over a decade old, almost two. There has been a noticeable change in how technology is used, and with the introduction of smartphones, the CPS system is rendered obsolete. That is to say, the physical device itself. The technique and activity itself are still in active use, even on

NTNU. The rise of active quiz programs like Kahoot [8] have been actively used on in classrooms for several years and is widely utilized worldwide, for both lower and higher education purposes.

2.2.5 IN1900 - Introduction to Programming with Scientific Applications

The IN1900 course at the University of Oslo (UIO) [21] is an introductory course for students with scientific study programs, such as physics and mathematics. The course's goal is to give students an introduction to programming by illustrating mathematical examples concerning programming. The learning outcomes section of the course description says that the student will learn "basic programming skills in python, learn to solve real minor issues on a computer with graphics (plotting, animations), and be able to create sketches and algorithms based on a mathematical specification of a science problem "[21].

From the course page and attached course material online [19], this course is based on wholly mathematical themes that are used as examples. It is specially created for students with a non-IT major in mind, which is related to the topics that this thesis is looking into for possible improvements of learning for non-IT major students at NTNU. It would seem that, since 2017, another university has divided students with similar discipline into different courses, based on if the discipline is an IT-based one or not.

It is also worth noting that the book mentioned in the course material, is written specifically with examples taken from "mathematics, numerical calculus, statistics, physics, biology, and finance "[28]. Since this course is used at UIO, there could be argued to find a similar solution for students at NTNU.

2.2.6 Computational Methods in General Chemistry

There have been previous attempts to look at teaching scientific fields through programming exercises or other programs at the University of Virginia. Wheeler and Lindsay et al. 2016 [33] looked into teaching chemistry students programming skills with a specific focus on solving chemistry problems that can often be very complex to do by hand. They found results that showed students being much more comfortable to solve and even create tutorials for solving chemistry problems, albeit in a confined environment of the Mathematica application from Wolfram [18].

The researchers give a reason for conducting this survey and test students in an engineering-specific course, CHEM1621, which is an introductory course for chemistry, specifically in a laboratory setting [16]. In later course years, and in specific fields of work like STEM cells research [33, p.83], there is a high use of

computational programming to solve problems, as well as plotting, simulation and more.

Although the use of a specified language developed for technical computing, there is some evidence[33, p.87] shown that students did have a better experience with solving problems in a programming related way. The paper points out that students with no previous programming experience did have more problems than experienced students, and thus, did not see the usefulness of using Mathematica in later courses. This shows that there would not necessarily be an advantage to use confined applications with new students to learn programming. The paper also points out that experienced students did have a better time with the application and had a higher end-result at the end of the semester.

2.2.7 Conclusion

To conclude the findings of this chapter, there is shown from several different source on both how one can implement an effective exercise program, as well as connecting programming towards non-IT students in a feasible way. There are both online oriented programs, such as Kahoot [8], as well as existing courses, like IN1900 [21] that has relevancy towards this thesis.

Chapter 3

Basic Theory

This part of the paper focuses on basic information regarding the thesis. It includes common terms, methods used later in the paper, as well as how the research will be conducted. There will also be a section to justify why the mentioned methods are used.

3.1 Methods

3.1.1 Data gathering

Questionnaire

The questionnaire is a quantitative method for gathering a large amount of data from a large focus group quickly [31]. The questionnaire consists of a pre-determined set of questions presented to the participant. The questions can be self-administered; the participant completes the questions themselves without any oversight from researchers, or researcher-administered. For this thesis, it was decided to administer the questionnaire as a self-administered, as the focus group of students that participates in the TDT4110 course consists of a wide array of different study courses. The questionnaire will be administered through different channels, such as lectures, among others. The questionnaire is dependent on the number of participants to gather a decent amount of data that is useful.

Some problems can arise from using a questionnaire. The main problem would be the indifference of potential participants. Since this is a non-binding questionnaire survey, no one is forced to participate in the survey and could cause potential and valuable information to be lost to the researcher. There is also a problem with participants being separated from the researcher, in that they would not usually see

the researcher in person, which commonly occurs with online questionnaires such as this one. There is also a need for the questionnaire to be clear and concise to understand and interpret the questions correctly, or else there may be misunderstandings between the participants and the researcher's questions.

Interview

From the questionnaire, there will be possible for participants to apply for an interview. This interview will be constructed to take a more in-depth look at the participant's view on the course, as well as additional questions regarding potential changes to the exercises of TDT4110.

The interview will be a structured based one, with live notes as well as recordings that will be transcribed. Interviews are suitable data gathering methods because of their ability to retrieve answers that are more complex and open-ended than questionnaires [31]. The questionnaire will create an overview of the student consensus of the TDT4110 course, while Interviews can explore a more narrowed approach to several questions that were relevant for the study.

While Interviews are an excellent tool for dealing with topics in-depth, several disadvantages might arise and needs addressing. There are often problems of reliability that concerns interviews. It is hard to achieve proper consistency and objectivity with the context and researcher having a potential effect on the interview object [31]. There is always a particular bias that will influence the researcher's questions, since they actively are working towards a research goal, and wants to achieve that goal in some way. There is also a problem of communication, where the researcher may say something, and the interviewee interprets as something else.

3.1.2 Choice of data gathering methods

To better show the researcher choices for these data gathering methods, the following section will contain reasoning and discussion of different methods.

The research questions for this thesis was defined in chapter 1 as:

- How can thematic exercises help improve students' performance in introductory programming courses?
- Can these tasks be, in some way, implemented into the existing system, and if so, can they replace the current exercises?

The research questions would need to be reflected correctly in the data gathering methods used in the thesis. The first argument for a questionnaire and a

series of interviews is the time constraints that concerns the thesis. This thesis has gone through several iterations since the project start, including a significant change in thesis description, and had significantly less time for conducting more time-consuming research. The following subsections describe alternative methods not chosen.

Case study with volunteers

A complete version of data gathering would be with a complete case study[31, p.141] lasting the majority of the semester. This method would most likely consist of a new exercise program with thematic problems to be solved connected to a primary scientific discipline, such as math, and two or more surveys to gather students' opinion and experiences. This would by far be the most complete and valid data gathering to utilize with a thesis such as this. The problem lies in two factors; the time needed to prepare such a project, as well as timing the data gathering period with the TDT4110 course.

Since a Master thesis for computer science is conducted over a year, starting in the fall semester, there is a problem with syncing such a project with TDT4110, as the course itself is in the fall semester as well. The researcher needs ample time to prepare for the case study, with exercises, surveys, and finding volunteers, which should take a large part of an entire semester. The time and resources needed to be put into such a project would also have a potential to go beyond the scope of a master thesis, and with the additional problems of syncing with the TDT4110 course, would make this kind of data collection unsuited for this thesis.

Experiments about motivation and learning effects

Having an experiment with a focus on motivation and learning effects, where one conducts a pre- and post-survey with two groups of students at two points, before experimenting with one group between the two surveys, before comparing results[31, p.126]. In this way, one could test out actual exercises that are thematically aligned with students' scientific field of study.

The main problem arising with a data-gathering system is, as mentioned in the previous section, time and synchronization with the semester and the TDT4110 course. By performing this kind of data gathering in the spring semester, one would miss out on both candidates, as well as fresh opinions from students. It would be unwise, as there is a much better environment to record student opinion on thematic exercises and other relevant topics in the fall semester.

On the other hand, one could collaborate with faculty members and lecturers to better promote the experiment to students that may be interested. This solution would better the process of testing students with new exercises, tests, or other

related work for gathering data. Working with faculty would also give a better insight for the faculty itself, if they are interested in the findings, and could be a stepping stone towards reiterate the TDT4110 course. Still, the timing, and also a large amount of preparation would have to be done before such an experiment could be completed. This shortened version would although be easier to complete, than a full case study, as the experiment could be limited in scope to a few sessions with students, compared to creating an entirely new exercise program for students.

Questionnaires and interviews over other methods

So why choose questionnaires and interviews over these other methods, besides the pros and cons mentioned above? One of the reasons is already mentioned time and resources. As this thesis was revised at the start of the second semester, there was no time to plan and execute more complex and more in-depth research. Finding volunteers to participate in the questionnaire was severe enough when one has to search through what courses first-year students in relevant study programs take in the spring semester and then plan to come in and advertise in a lecture to gather data. Having done this during a lecture in TDT4110 would have been much more straightforward, and most likely give a better result than the solution given.

Also, the purpose of the thesis was not to develop and replace an existing system of exercise, but instead find out if there are a valid advantage and willingness for non-IT students to change the system and as a result, give students better skills to utilize later in university and beyond.

3.2 Ethics

With all research, there is a clear and important focus on ethical research. There are restraints, rules, and laws that need to be accounted for with all research [31, p.55]. In this project, the primary concern is the privacy of participants in the interviews and questionnaire.

From the web-pages of the Norwegian National Research Ethics Committees [15], The respect and assurance for the integrity of an individual are paramount for ethical research:

> Respect for human dignity and personal integrity is formalized and laid down in a series of international laws and conventions on human rights[2, §102]

There is simply both international and national law that prohibits the potential exploitation of an individual's integrity if a person participates in any form of re-

search as a 'subject.' This project is therefore obliged to follow these rules and guidelines to gather data from individuals that will participate in the project.

In recent years there has been an increased focus on the digital world, with the GDPR guidelines implemented in 2016 [7]. The law lays new protective laws on top of the existing ethics laws and guidelines that already exist for researchers, and while the law focuses on general consumer rights and protections on the internet, it also applies to research data stored in digital data-centers. This project will save potential personal information from participants.

3.2.1 NSD

The questionnaire will store information that can potentially be used to track back to the participants. To protect the participants' personal information, the research project has applied to the NSD (Norwegian Center for Research Data) that gathers, stores and reviews all parts of research projects that will handle personal information and other research data [12]. See appendix (INSERT APPENDIX PAGE HERE) for the application forms.

3.2.2 Storage of personal data

The storage of personal data will be locally on a computer provided by the institute. The computer is only accessible for the researcher and admin personnel. The data will also be intermediately saved on the Microsoft Forms application provided by the university and deleted when the questionnaire closes.

3.2.3 Personal information

Personal information is defined as any information that can be related to any individual[14], such as name, addresses, a telephone number, or e-mail. There is also other pieces of information that may be connected to a person, such as a gender, behavioral patterns, and IP-addresses.

The personal information gathered is minimal and will, in practice, not be able to be tracked back to any participants of the questionnaire. The sensitive data gathered are participants that wish to be interviewed in another part of the research project.

The main concerns of personal privacy are the gathering of Gender information, study program, and E-mail. The first two will, in isolation, not be able to determine the identity of a person. If the two are connected, with the addition of E-mail addresses, there is potential to connect to an individual. These pieces of information are therefore locally stored so that no one else but the researcher

can access this information. In relation to the interview, audiotapes are considered personal information, even though there are no names related to participants mentioned during recording. These recordings will be saved locally to a personal computer that only the researcher has access to, and will also be deleted when the project is delivered.

Information	Defined as personal information?
Study program	Potentially yes
Gender	Potentially yes
Year of study	No
E-mail	Yes
Audio recording	Yes

Table 3.1: Table of information that is gathered by the questionnaire and interview

3.3 Other

3.3.1 TDT4110 - Information Technology, Introduction

The main focus of the thesis concerns the course TDT4110 - Information Technology, Introduction [4]. This course is the basic introduction course at NTNU that has the main goal of introducing new students to the information technology (IT) discipline and teaches the fundamental concepts of IT. The thesis focuses a great deal on this course, as it is a course that many study courses outside of the main IT-discipline are required to complete as an introductory course on their first or second years.

3.3.2 Microsoft Forms

The Microsoft Forms program is a part of the Office 365 package that students and teachers at NTNU can access through the University's intranet [13]. The form is highly flexible and supports multiple types of questions. The main reasoning behind this choice of system is that the data gathered is saved on internal teams that only the creator or administrators of the form can access the gathered data unless explicitly shared by the original creator. By utilizing this system, the data can be safely stored temporarily without the risk of any third parties that can access it.

Chapter 4

Survey

This part of the paper will contain the methods of data gathering, the survey results, and notable findings from these results. There will also be a summary and analysis of several interviews with students from non-major study programs that have completed TDT4110.

4.1 Introduction

The survey was conducted with the use of the Microsoft Forms survey tool that was available at the university's internal websites, as mentioned in Chapter 3, while the interview was conducted face to face between the researcher and voluntary participants.

4.1.1 Distribution

The questionnaire was distributed by URL-link verbally communicated in lectures for different study programs. The total lectures in which the URL was distributed in totaled to 7, with additional distribution on the course web pages on the Blackboard intranet, with the help of several professors and other lecturers. The interview invitations were also indirectly distributed with the questionnaire. The questionnaire also included an additional text space at the end of the questionnaire, where students could send in their e-mail address if they wanted to participate in the interview.

4.2 Structure

In this part of the paper, the structure and content of the questionnaire and interview will be presented and discussed.

4.2.1 Questionnaire structure

Part 1

The questionnaire was structured with a three-part division in mind. The first and shortest part was the gathering of general information about gender, year of study, study program as well as any previous experience with programming before taking the TDT4110 course. The main idea of this part was to see any differences from each course during the analysis of the questionnaire. How does a physics student differ in programming from a machine student? Is there a difference between female and male respondents? How will a responder from a higher year differ from a first-year student? Every question was asked to compare the results of these basic information questions with the rest of the survey.

Question:	**Notes**
What is your gender?	Male/Female/Other response
What year are you in?	Choose between 5 options (1-5)
What is your main study program?	Drop-down menu
Have you had any previous programming experience before beginning your study program?	Yes/No response

Table 4.1: Questions and response options used in the questionnaire

Part 2

The second and central part focused on the participant's experience with the course itself, noting the students' enjoyment, the feeling of difficulty, and what part of the curriculum was easier or harder. These questions were structured as a preference scale along a horizontal line, with values 1-5. The first two questions regarded the difficulty and enjoyment of the course before the participant was tasked with rating each of the primary topics in the curriculum. This section then asked participants to answer if they were considering taking more programming courses during their education and if they thought that the course would be a benefit for their study program and discipline.

There is worth noting that the rating scale differs with it values and what they represent in some cases. The first question in table 4.2 ranks the difficulty from 1 to 5, as the value increases, the difficulty also increases. The second question about enjoyment, and the second to last about IT and benefit towards one's study program rates the other way around, with the 'positive' value at 5, and the 'negative' value at 1.

Question	Note
On a scale from 1 to 5, how difficult would you rate the TDT4110 course?	Scale 1-5, from very easy to very difficult
On a scale from 1 to 5, what is your overall enjoyment after completing the TDT4110 course?	Scale on 1-5, from not very enjoyable to very enjoyable
The below questions are concerning your performance on individual parts of the course. On a scale from 1 to 5, how would you rate the difficulty of each aspect of the course:	Scale 1-5, from very easy to very difficult. This was asked in a table of ranking scales
On a scale from 1 to 5, how beneficial would you say programming experience will help you in your main discipline?	Scale 1-5, from "not beneficial at all" to "Very beneficial"
Are you thinking about taking any more courses related to IT during your study program?	Yes/No response

Table 4.2: Table of question from the second part of the questionnaire

Part 3

The last part of the questionnaire focused on the introduction of thematic exercises into the course and how students would respond to these ideas. These questions focus on the student's feelings and opinions on implementing a thematic option for the exercises in TDT4110. This part pitches students opinion against each other, and of what preference they have towards either of the presented topics.

There are three topics presented: The participants own study discipline (Mathematics, Chemistry, Civil Engineering), Generic tasks, i.e., the typical tasks one would face as part of an exercise in TDT4110, and lastly, Mathematics. The last topic is based on that all study programs on NTNU include some form of mathematical course in the first semesters [24]. There are also three additional questions in this section. The first questions the participant if they would like to see themes

from their study program implemented as exercises in TDT4110. The second is an open-ended question with a non-mandatory text field where participants can enter any additional thoughts and feedback on the course. This question was added for having an additional method of gathering feedback and opinions from participants that would otherwise not be answered from the rest of the questionnaire.

There was also an additional field for entering e-mails. This field was added to recruit participants to the interview later in the study.

Question	Note
On a scale from 1 to 5, how beneficial would you see the implementation of related course material from your own special field of study?	Scale from 1 to 5
How is your preference between completely generic tasks and tasks related to your special field of study?	Scale from 1 to 5
How is your preference between completely generic tasks and tasks related to mathematics?	Scale from 1 to 5
How is your preference between tasks related to mathematics and tasks related to your special field of study?	Scale from 1 to 5
On a scale from 1 to 5, How willing would you be to participate in a demo with a few programming exercises tailored to your discipline?	Scale from 1 to 5
If you have any further feedback or opinions on the TDT4110 course or the questionnaire, you can write them down below	Open comment for those who would add additional feedback
If you want to participate in an in-depth interview, please enter your e-mail below.	Non-obligatory field where participants can enter their e-mail

Table 4.3: Table of questions from the third part of the questionnaire

4.2.2 Interview Structure

The interview was conducted with voluntary participants from the digital questionnaire. They were able to voice interest in the interview by entering their e-mail addresses at the end of the form and were contacted at a later date for more information. The information mail distributed to responders contained information about what the interview entailed, the estimated duration of the interview, as well as assurance that any private information about participants was to be kept strictly

confidential, in line with research guidelines and privacy laws of both national and international scope. The information was distributed in Norwegian and English so that international students had an opportunity to participate in the interview.

The interview itself was designed to be short and concise so that there was a more significant chance of attracting willing participants to interview. The length was estimated to be about 10-12 minutes long, with a leeway duration of about 3 minutes over the estimate. Extended interviews are often tedious for both the interviewer and interviewee, and since the participants are students with busy schedules, there was a clear focus on keeping the questions on the topic and relatively short.

The structure of the interview itself could be defined as a semi-structured interview [31, p.188], where the questions are pre-determined but are flexible enough so that the interviewer can change the order that the questions are asked, depending on the flow of the conversation. There was also room to ask additional questions if there were needed. This freedom also allows the interviewee to be more detailed about topics and issues raised in the interview. This also raises some issues that the interviewer needs to address, namely that the interviewee will have more freedom to talk away about one issue and risks going off-topic if one is not careful to steer the interview accurately. Oates argues that semi-structure is a method that has a clear goal of 'discovering' as the primary purpose. This method fits this research paper very well, as the purpose here is to determine and discover what students think about implementing thematic exercises into TDT4110.

Below is a list of the main questions asked:

1. What is your opinion on the TDT4110 course? What did you like? What did you not like? What could have been done better?

2. How do you usually progress when solving a programming exercise?

3. You will now be shown 2 different programming exercises. One is derived from the generic programming exercises in TDT4110. The other is an exercises created from a theme that is in line with your own study program. You will be tasked with reading through the exercises and then make a decision on which one you would rather had have as an exercise on the TDT4110 course.

4. What is it specifically you would have liked to see be a part of the exercises for TDT4110?

5. What other courses did you take parallel with TDT4110?

6. What do you use the knowledge you attained from TDT4110 to now (in relation to other courses, jobs etc.)?

The questions were primarily focused on the students' experience and wanted to retrieve more details from the participants than the questionnaire did. There is certainly a limit to how one can rate their enjoyment and feelings on difficulty with one number between 1 and 5. This scale allowed the interviewer to retrieve the more specific reasons for their ranking scales from the questionnaire. There is an opportunity to get specific answers on what was the most challenging part of the course, what the easiest one was, and what the participants felt about the course in their personal view.

There was also an opportunity to ask about their habits and how they solved a programming exercise. How do they progress? Is there a difference between participants, and if so, do this relate to their study program and study field? Is there a difference in how students from Biochemistry solve tasks in comparison to students with specializations in civil engineering? These questions also relate to the existing background literature mentioned in Chapter 2, with the added benefits of solving problems with computational thinking, and how to apply these methods in other fields.

Thematic exercise demo

The main activity that takes part in the interview is the student's decision on the third question. Since the research was interested in how students would react to programming exercises with a theme related to their main study program, the master student developed a series of programming questions that derived from both the existing exercises in TDT4110 and other disciplines. Since there was a limit on how many different version that the researcher was able to create because of time restraints, there was a decision to focus on the three main scientific topics: Mathematics, Physics, and Chemistry. All students with a focus of some scientific study will include this in their study program. Mathematics is especially prominent, featuring in all study programs.

The structure of the exercises was built to be nearly identical between all tasks included. The reason behind this decision was to align the exercises to be as similar as possible so that there would be no noticeable difference in difficulty between standard exercises from the TDT4110 course, and exercises crafted by the researcher.

The researcher wanted to explore students opinion on the introduction of their discipline into the TDT4110 course, and since there was a concern that the student would possibly misunderstand or lack some insight into what the purpose of the study was, the "demo" was included into the interview where the student would

be tasked with a short read-through before voicing their opinions on what exercise they preferred.

Programing task - MATHEMATHICS

You are going to make a basic program that will take a temperature reading in Celsius (C) as an *input* from the user, and then convert the temperature to Fahrenheit (F). The conversion formulae are as follows:

$$F = \frac{9}{5}C + 32$$

The output should show the original temperature input, as well as the output after conversion.

Part 2:

You are going to expand upon the previous program, and check if the result (Fahrenheit) is above or below 0. Print a fitting response for the result.

Part 3:

You are now going to program a series of conversions, using *loops*. The goal is to start by converting -20 °C and then increment by 5 °C until you reach 40 °C. Each conversion shall be printed out with both the Celsius and Fahrenheit results.

Figure 4.1: Programming exercise with Math-theme

Programming tasks - IT (retrieved from the TDT4110 course)

Part 1:

You are going to take a string input from the user, asking for their name. The program will then return the name with a fitting sentence. For example:

> "What is your name?": John
>
> "Hey John, cool name!"

Part 2:

The goal of this exercise is to make a program that checks if the user can vote. Use an input that takes an integer and checks if the user is above voting age, that is, 18 years old. Print a fitting response if the user is above the minimum voting age.

Part 3:

You are going to make a program that, by the help of loops, are going to ask the user for 7 numbers. The program is then going to sum all of these numbers, before returning the result to the user.

Figure 4.2: Programming exercise with IT-theme

Questions regarding participants current use of Information technology

After the "demo" concluded, the remaining questions focused on the participants use of their knowledge in their area of study. Was programming or other points of the TDT4110 curriculum utilized in their current courses, or were they aware of future courses that did the same? If the researcher could retrieve a clear overview of the participants need for these skills, and if any special topics were missing from

the current curriculum in TDT4110. The participants were asked if they used these skills in other areas than for educational purposes, such as projects on their free time or part-time jobs.

4.3 Initial results

Questionnaire

The result from the questionnaire ended at 166 participants, of which 136 was first-year students, 20 was second-year students, and the remaining ten was in a higher year. Gender participation was a stable 47% women and 53% men. Four groups stand out with a notably high participation rate and should be examined. The main bulk of the results came from Chemical engineering and Biotechnology students with a large lead of 52 participants. There was also a large presence of students from Mechanical Engineering (MTPROD) with 27 participants, followed by various participation from Applied Physics and Mathematics (MTFYMA) and Civil and Environmental Engineering (MTBYGG). Table 4.4 shows the different participation, divided into study programs and gender distribution.

Participants did not widely utilize the questionnaire's 'additional comment,' all though there were some comments about the course, and it is shortcomings. Of the 11 comments, eight focused around the lack of plots and graphs in the curriculum. There was also one participant that did not understand which faculty that was responsible for the TDT4110 course, believing that the Faculty of Natural Sciences was responsible. The fact that there is at least one person that has the wrong impression of who has the responsibility may imply that there are several students that share this view, and as such, has the impression that their faculty do not do enough to facilitate their students when it comes to Information technology skills.

Interviews

From the questionnaire, there were a total of 18 participants that were interested in doing an interview. Of those, there were five that said yes to the offer. The interviewees were less varied than the participants in the questionnaire, with four from Mechanical Engineering, and one person from Applied Physics and Mathematics. Every participant was in their first year in their studies. Participants were compensated with a gift card.

Study Course	**Number of participants**	**Gender difference (M/F)**
Chemical Engineering and Biotechnology	52	34,6%/65,4%
Applied Physics and Mathematics	20	45%/55%
Civil and Environmental Engineering	18	38,8%/61,1%
Mechanical Engineering	27	18,5%/81,5%
Materials Science and Engineering	13	53,8%/46,2%
Natural Science with Teacher Education	11	45,5%/54,4%
Industrial Economics and Technology Management	8	37,5%/62,5%
Physics - Bachelor	6	33,3%/66,6%
Other	5	60%/40%
Chemistry - Bachelor	1	0%/100%

Table 4.4: Table of participants with gender difference

Chapter 5

Analysis & Discussion

In this chapter, the findings from the questionnaire and interview will be analyzed. A discussion will follow this analysis before stating threats to validity.

5.1 Questionnaire analysis

5.1.1 General findings

The first part of the data is basic information about participants. On the question of previous programming experience, there were over 77 % that answered no. 128 of the 166 participants had not had any form of programming experience before starting the course, a wast majority, if one correlates these participants to the general populace of first-year students. Approximately 2000 students take the exams each year [9], in which the 166 participants would make out an 8,3 % of the assumed total that completed the course and received a grade. The margin of error, with a confidence level of 90 % would land at about 7,2 %, which is well inside the margins of error.

There is a clear divide in the number of participants, with a large number studying biochemistry. Over 50 participants were recorded. The Other large groups are civil engineering, Applied physics and Mathematics, and Mechanical Engineering (see 5.1 for a full breakdown of participants' study program).

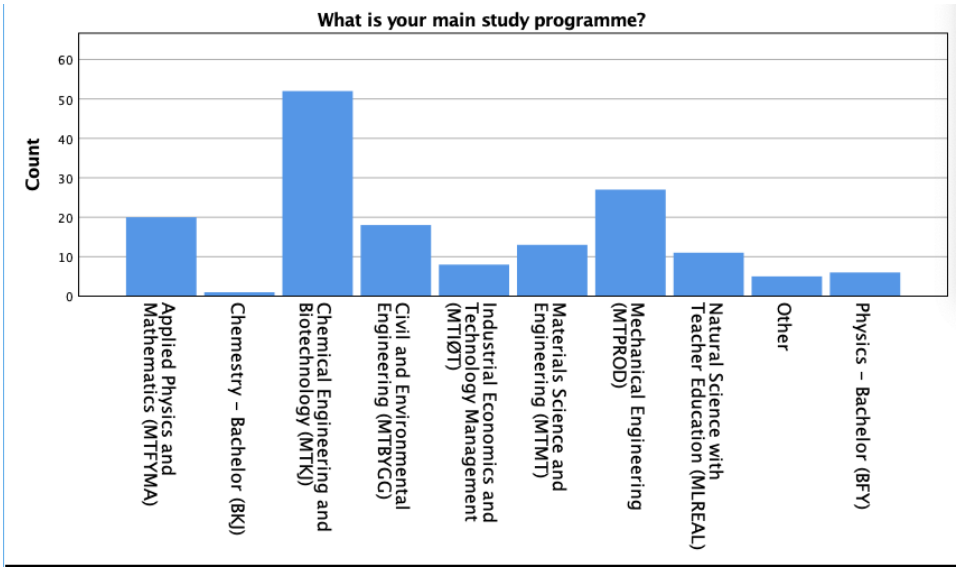


Figure 5.1: Chart over student participation in questionnaire

Difficulty and enjoyment

At first glance, Biochemistry participants answered that they had more difficulty with the TDT4110 course than the questionnaire average. With over 73 % answering a 3 or 4 on the question: ”On a scale from 1 to 5, how difficult would you rate the TDT4110 course?”. When comparing, the same cumulative percent in Mechanical Engineering shows that the majority of participants answered 2 or 3 on the same question, with over over 80 % answering 3 or below. The same can also be noted for Applied Physics and Mathematics, with similar results. Civil and Environmental Engineering also has a skewered divide of answers, with over 50 % rating the difficulty of the course as a 4, with no one answering 1. Table 5.1 shows the answers of each grouping, divided into percent groups. There is thought to look more into Civil and Environmental Engineering and Biochemistry and how they compare with each other and the other large groups.

Difficulty:	Biochemistry	Mechanical Engineering	Applied Physics and-Mathematics	Civil and Environmental-Engineering
1	3,8	3,7	10	0
2	15,4	40,7	70	27,8
3	34,6	44,7	15	11,1
4	38,5	11,1	1	50
5	7,7	0	0	11,1

Table 5.1: Participants rating on difficulty in percent (%)

However, if one compares the average means of each study program with each other, it shows that the highest difficulty rating lies with Materials science and engineering (MTMT) with an average mean of 3.85, which is significantly higher than what Biochemistry lies at, with 3.40. This result has most likely to do with the number of participants divided between the two study programs, and that the result may be different if there were more participants from Materials science and engineering. Table 5.2 shows an average mean difficulty overall study programs

What is your main study programme?	Mean	N	Std. Deviation
Applied Physics and Mathematics (MTFYMA)	2,15	20	,671
Chemestry - Bachelor (BKJ)	3,00	1	.
Chemical Engineering and Biotechnology (MTKJ)	3,31	52	,961
Civil and Environmental Engineering (MTBYGG)	3,44	18	1,042
Industrial Economics and Technology Management (MTIØT)	3,38	8	,518
Materials Science and Engineering (MTMT)	3,85	13	1,068
Mechanical Engineering (MTPROD)	2,63	27	,742
Natural Science with Teacher Education (MLREAL)	3,09	11	,831
Other	2,40	5	,548
Physics - Bachelor (BFY)	2,00	6	,632
Total	3,02	161	,997

Figure 5.2: Table showing mean average of difficulty, divided by Study program

There would be preferable to look at the relationship between difficulty and enjoyment of the course, and its content. By comparing the overall difficulty and enjoyment question asked in the questionnaire, one can get an overall state of

TDT4110 and how students see the course content. One of the solutions to answer this question would be to use the Spearman correlation formulae to see if there is a negative or positive correlation between enjoyment and difficulty. This formula is performed as taking the Spearman formulae and applying it to the results [26]. The following table shows the results of using the correlation formulae:

			On a scale from 1 to 5, what is your overall enjoyment after completing the TDT4110 course?
Spearman's rho	On a scale from 1 to 5, how difficult would you rate the TDT4110 course?	Correlation Coefficient	-,516**
		Sig. (2-tailed)	,000
		N	161
	On a scale from 1 to 5, what is your overall enjoyment after completing the TDT4110 course?	Correlation Coefficient	1,000
		Sig. (2-tailed)	.
		N	161

**. Correlation is significant at the 0.01 level (2-tailed).

Figure 5.3: Correlation between difficulty and enjoyment

The table, taken from the SPSS analytic program, states that the correlation between the difficulty and enjoyment has a - 0,516 correlation coefficient with a p-value = 0,00 (Sig. 2-tailed), meaning that there is a relative strong relation between the two questions. When a person has answered low on the difficulty rating for the course, it is more likely that they would have a higher enjoyment with the course. However, it could also go the other way around. If a student with a large amount of previous experience thinks that the exercises or other parts of the course are too easy, they could end up being bored or uninterested in the broader aspects of the course. If this happens, they could miss out on some areas that they may not have learned earlier, or have their enjoyment of the course lowered as well. This phenomenon is centered on the theory of Flow, from Csikszentmihalyi and Rathunde [22] in the field of psychology. If a person finds a challenge either overly challenging or overly simple and easy, they often result in boredom and little to no enjoyment.

Perceived benefit of programming

Several of the questions were directed towards participants to get a better overview of how students perceive information technology as a beneficial tool in their pri-

mary discipline. As explained earlier, there is an argument for having general understanding and knowledge of programming, and computer science across several fields of study.

			On a scale from 1 to 5, how beneficial would you say programming experience will help you in your main discipline?					
			1	2	3	4	5	Total
Have you had any previous programming experience before beginning your study programme?	No	Count	1	19	35	49	24	128
			0,8%	14,8%	27,3%	38,3%	18,8%	100,0%
	Yes	Count	1	1	10	5	16	33
			3,0%	3,0%	30,3%	15,2%	48,5%	100,0%

Figure 5.4: Table shows responders' rating on benefits of programming, crosstabed with their answer to the question regarding previous programming

The questionnaire asked several questions regarding participants' relationship with programming and how they perceive the usefulness. 5.4 shows a cross tabular that compares answers from two different questions:

- Have you had any previous programming experience before beginning your study programme?
- On a scale from 1 to 5, how beneficial would you say programming experience will help you in your main discipline?

The vast majority of participants said they had not had any previous experience with programming before attending the TDT4110 course. Over 128 (77,1%) of the 166 participants answered "No" to any previous experience, while the remaining 33 had. What is also interesting to note is the difference between the rankings of benefits between the "No" and "Yes" groups. Of the participants that answered "No," there were 57,1 % that ranked the benefits of programming as a 4 or 5, while 63,7% in the "Yes" category would rate the benefits at the same level. Breaking the ratings further down, almost 50% of the participants in the "Yes" group would rate the benefits of programming as a 5, while only 18% of the "No" group would rate the same.

This result could come from the fact that one usually would understand a scientific field better when one has more knowledge over a person that has not had any previous endeavors in the same field. The majority of students that starts in the first semester has, as the questionnaire shows, had little to no experience in the computer science field before. The field is different from the more classical fields of science, such as math or physics, and often requires a different method of thinking i.e., object-oriented and procedure-oriented paradigms. Students with previous experience could, very likely, understand and apply programming theory and skills better to their field of study rather than students with no previous experience.

One should, however, note that, since the number of answers regarding previous programming experience is skewed between participants, there could also be missing data from the main student body that did not participate in the questionnaire, and that the figures could be more distributed among the ”perceived benefit”-ranking if more people answered.

Plans for future programming courses

There is also worth looking at how participants look to the future of their programming knowledge. There was asked a question: ”Are you thinking about taking any more courses related to IT during your study program?”. This question is a simple ”Yes/No” question that looks at students further plans to educate themselves more in the IT fields of science, even though they are not majoring in the discipline.

		Are you thinking about taking any more courses related to IT during your study programme?		
		No	Yes	Total
Have you had any previous programming experience before beginning your study programme?	No	60	68	128
		46,9%	53,1%	100,0%
	Yes	9	24	33
		27,3%	72,7%	100,0%

Figure 5.5: cross-tabular of participants' plan on continuing with programming courses

This figure shows a simple cross-tabular that compares experienced and inexperienced students with their answers to taking further IT-courses. There is a clear majority of experienced students that they want to take one or more courses that relate to IT. There is also an almost equal divide between the inexperienced students if they want to do the same. There is also the fact that students have different needs for their study programs. A chemistry student may not use the same programs, functions, or even the same programming language as a mechanical engineer. They may also have a different amount of usage for programming, with a biology or chemistry student would need to utilize programming for visualization or simulation, while a mechanical engineer would use the same set of skills for use in robotics, or other hardware-focused areas. As such, there may be a different demand for different study programs to take more programming or IT related courses, depending on their specific field.

Topic difficulty

As the participants were tasked with rating the different chapters and topics in the TDT4110 course, there would be beneficial to take a look at the results. Seeing what participants regard as difficult, would benefit the overall study, as it would reveal where students struggle in the course.

Thematic preferences

Participants were asked to show their preferences towards several academic themes and how they would compare each topic towards what was defined as ”generic tasks.” Mathematics, generic tasks, and the participant's field of study were compared with each other.

	No previous experience	Previous experience	Total average
How is your preference between **completely generic tasks** and tasks related to **your special field of study**?	3.29	3.42	3.32
How is your preference between **completely generic tasks** and **tasks related to mathematics**?	3.12	3.15	3.12
How is your preference between **tasks related to mathematics** and tasks related to **your special field of study**?	3.27	3.73	3.36

Table 5.2: Table with preference average for student with and without previous programming experience. the scale is from 1 to 5, where edges are extreme preferences towards one option and the middle (3) is no preference. extreme edges are in bold letters

From the 5.2 Table shows the average response from participants, divided between participants with and without previous programming experience. There is not much of a difference between responses here, although many say that they do not have any specific preference towards either option when presented with the three questions. There is a 3.73 average from participants with previous experience say they have a very slight preference to their particular field of study, rather than mathematics. Mechanical engineering has a higher average than Biochemistry, with 6 participants rating 4 of the total nine valid participants.

Overall, the average shows that there is little to no variation in participants answers, which could point to the questions being rather vague or poorly composed to participants. There would be preferable to have had more participants with previous experience so that the sample size would be more balanced, and as such, could have given a more detailed insight into student preference.

The questions themselves could also have been more straightforward and more transparent. There was an accompanying description for each question that needed an explanation, but there is no clear result that students have had a clear understanding of the questions. There is also the question of question type, with the 1-5

scale used in the questions. There may have been a better option to use a broader scale, with either 1-7 or 1-10 as a scale, to give the researcher a more nuanced result.

5.1.2 Other notable discoveries

In the questionnaire, the participants were asked questions, such as; ”On a scale from 1 to 5, How willing would you be to participate in an exercise example with a few programming exercises tailored to your discipline?”. The average response rate here was under average, 2.55, which states that students are not exactly excited with participating in an exercise example of the suggested content. This result shows that students need more incentive to participate in example-versions of new exercise-programs. One solution is to subject students to a trial version as a direct replacement for the existing programming exercises or offer them an extra curriculum for practicing to exams. Students were offered rewards in the form of cinema-tickets, but these did not seem to encourage students much. If the reward had been more substantial, there might have been a bigger turnout.

There was also allotted a voluntary possibility for participants to comment on the questionnaire. There were several that commented that the course lacked any topic of graphical design that could assist students, like plotting and graph simulation in python. One Mechanical engineering student commented that they felt that a crucial tool that was part of their study program as well as in mathematical relations to ”drawing surfaces and similar figures that were relevant.”

5.2 Interview

The interviews were performed on 5 participants, as there was neither time or resources enough to interview a large number of participants. There was five interviewees total, with four from mechanical engineering and one from Applied Physics and Mathematics. They were subjected to about 15 minutes of questioning, as well as a short exercise example, that consisted of stating their preferences towards different programming exercises.

General feedback on TDT4110

The majority answered that they thought the TDT4110 course was fun and enjoyable, with a good and basic introduction for new students to programming. The last participant was adamant that they did not have any previous experience with programming, and was quickly overwhelmed by new topics throughout the early weeks of the course, saying:

I felt that the course was pretty basic during the early exercises, but I felt that there suddenly was very many new topics in quick succession after we started with loops

The previous four all had a varied range of previous experience, from basic understanding to large amounts of expert knowledge about programming. There were also some comments from the experienced participants where they felt that much of the content in TDT4110 was very basic and uninteresting for them, where they did not learn anything new. Since the course is directed to giving all students a fundamental understating of programming, it would make sense for experienced students to quickly complete the course material and show low interest towards participating in lectures. Introducing specific, thematic problems to students throughout the course could, therefore, lead to experienced students not learning these mechanics or how they relate to their field of study.

There were, however, a broad agreement between participants that thematic exercises would be an excellent addition to TDT4110, as it would help them prepare towards later courses that took use of programming. In both Applied Physics and Mathematics and Mechanical Engineering, several following courses require at least some basic understanding of programming. Having a relation between these courses, or general topics and programming would be valuable, they said.

Useful topics missing from the curriculum

On a more specific topic, the participants were asked if they felt that something was missing from the curriculum in TDT4110. They answered that they felt there should be a basic introduction to plotting, because of the large number of courses that took use of computer-generated plots concerning exercises. Both Applied Physics and Mathematics and Mechanical Engineering would make use of these, as there was a real need for such tools in courses, the next semester. This addition to the curriculum would be of help for students, as the interviewees were not satisfied with the introduction to plotting that was given in the relevant courses.

By giving plotting, and other relevant techniques that are useful for students, as topics in TDT4110, would free up time in later courses and broaden the knowledge of the more extensive student body. One problem that could arise in this case is the chances of student assistants and lecturers having to include and broaden the current curriculum, as well as taking time out of the regular lectures that teach underlying themes to include what some may see as more advanced topics.

Assistance

When asked about where students tended to get help and support from, there was a wide array of answers. Some students used friends with a focus on solving exercises together. Many of the participants used the internet as a source of information, while others required no external help. There was no widespread use of student assistants from any of the participants. Students generally have widespread use of internet resources to make use of, since there is generally an immense amount of information that is easily available in comparison towards student assistants.

Solving problems

The interviewees were asked to describe their approach when solving programming exercises and problems. Some used a step-by-step process to solve tasks, while others used an iterative process. From the interviewees, there was no identical process, and they each had unique strategies for solving problems. This observation would make sense, as there is some obvious difference in the amount of experience each interviewee had with programming. Also, since every student most often has a different approach to solving problems, there would be strange to have every interviewee taking a similar approach to exercises.

Perceived usefulness of programming

The interviewees were also asked to perceive how useful the programming skills they acquired were going to be concerning their future courses in their study program. There were a perceived consensus from the participants that both Mechanical engineering and Applied Physics and Mathematics. Interviewees said that they would have an easier time with their future courses with a broader knowledge base in programming for specific tasks. There was a consensus that plotting was something that was repeatedly coming up as an example. Robotics was also named as a useful area to know programming, although, at a more complicated and machine-near level, which is a topic not covered in TDT4110.

5.2.1 Exercise examples

When presented with the exercise example, the interviewees were presented with a sheet of paper with two tasks they were told to study for 30 seconds to 1 minute. Afterward, they were told to present their preferences to either one of the tasks.

Machine engineering Interviewees

One interviewee presented their preference to the math tasks and told of the increased presence of using formulas in python programming to solve exercises or plot in different graphs for related math problems. There were pointed out that several exercises in mechanical engineering courses were often delivered with formulas that were then added to a program to calculate answers. There was also a comment on the number of strings used in the IT exercise, where some said that it was uninteresting and not relevant for them to have such exercises.

There was also made arguments for keeping with the current schedule. One of the more experienced interviewees made a point towards having more simple and general themes in the exercises since there are a lot of new and inexperienced students that are not comfortable yet with more mathematical exercises. Having more general exercises first to build a fundamental knowledge base for the students is certainly a valid argument that needs to be considered when thinking of changing the course. One solution would be to divide the course into two parts, with a more general section first, before dividing and specialize the second section towards a more nuanced theme that fits with the students' study program.

Applied Physics and Mathematics Interviewees

With the one interviewee from Applied Physics and Mathematics, there was not much further information that came to light from the interviewee, as there was not much of a different opinion given to the exercise example. There was a clear preference towards the mathematical version of the exercise and said that the string exercise seemed dull and not relevant for the interviewee to learn from, while the math exercise looked much more like an exercise that one could make use of in other math courses. The interviewee also agreed about parallel courses with TDT4110, having Mechanical physics in the same semester, that could have parallel exercises with TDT4110. This topic would need to come later in the course material, as there would need to be a basic understanding of the course material in Mechanical Physics first.

5.2.2 Further discussion

A further look at the results and analysis done from both the interviews and the survey finds that the general target group, early year students, would appreciate having more thematic exercises in the TDT4110 course. That would be a rather straightforward answer, but the inherent problem lies with how one would implement this as a practical course. Participants say, from the survey that they are reluctant to participate in an example version of the exercise program. There is

also the problem that lies in how specialized such a course would become, and how ample resources would have to be applied to changing the course to fit with each study program.

Dividing the TDT4110 course based on scientific areas

If the course exercise program is divided based on individual study programs, there is an apparent resource problem in that there would be a real problem to get enough student assistants to grade and return feedback to students. Since teacher assistants are hired from the student body at NTNU, there would be a severe lack of broader knowledge and expertise among these assistants in the case of grading exercises based on a study program that the teacher assistants belong to.

Because of this, there could be a solution to divide the study programs into different disciplines, making use of a typical scientific area that several courses could relate to, such as math or physics. From there, it would be easier to create a more specialized exercise course for students with similar courses, as well as pulling from a pool of teacher assistants that have the appropriate knowledge towards these main disciplines. It could also be argued to include more advanced programming tools, such as plotting, in these sub-courses to better prepare students better for later courses. Of course, these sub-courses have a trade-off, in the fact that more resources need to be allocated to this, in comparison to the existing lectures. More lecturers, teacher assistants, and a revised set of lectures are only a few problems that need addressing.

There is also the primary concern on how one would create these courses, and it would be profitable to make use of experiences from students in later years, as well as other lecturers in related courses that could give their own opinions in creating useful and related exercises that students would learn from. There would, therefore, take time to properly create an exercise system that can improve learning for non-it students, most likely taking several iterations before having proper and relevant course material implemented.

Non-obligatory exercises

Another possible solution to introduce ”themes” is to make extra, non-obligatory exercises and advertise these to non-IT students. This solution is a low-resource and cheap solution to expose students to related course material that may help them in later courses, and also exposes the problem of students not taking a proper interest in looking at the material, because of the non-obligatory state these exercises will be labeled under, and will as a result, not learn anything new. This solution could, however, be a possible stepping point for implementing thematic exercises as a necessary part of TDT4110.

Conclusion of discussion

To conclude this discussion section, there seems to be an area for improvement with having more thematic exercises that align with other study programs and scientific areas, such as maths and physics, but the means of implementing this, would need to be more closely investigated and tested on a deeper level than the survey and interviews discussed above in this chapter. The main problem that would have to be solved is how. How would one implement such a system, and in what capacity, where students would be properly engaged with the material in a meaningful way that can improve their understanding of how they can make effective use of programming to their benefit.

5.3 Threats to validity

This part of the thesis will state eventual concerns and possible errors in the findings. These points will be discussed and analyzed in short subsections bellow. The section will be divided between the questionnaire and the Interview data gathering methods.

5.3.1 Questionnaire

For the Questionnaire part of the data gathering, there are several concerns to validity. The first and foremost would be that students did not answer truthfully on the questions. This could be the participant simply becoming uninterested in the questionnaire and wants to finish the survey as fast as possible.

There is the chance that some of the students answering are not actually in the targeted focus group of the thesis. Since TDT4110 is a basic introduction course at NTNU, everyone that is a student at NTNU is allowed to take the course. That means if a student with a non-technological or scientific study program takes the course and participates in the questionnaire, it will introduce invalid data to the thesis, that could skewer the answers. The chance of this happening would be rather slim since the questionnaire was advertised in specific classes that would have a minimal chance of containing students with no relation to the focus group.

There is also the potential problem of selection bias contained in the questionnaire. Since there is a very variable amount of students from different study programs that have participated, the results is a skewered participation, with a majority of those that answered the questionnaire were from the Biochemistry study program, while, for example, Physics bachelor was poorly represented. Since these results are skewered, there will always be some leaning towards the preferences of

a specific study program, instead of having a more balanced distribution over several others.

5.3.2 Interviews

When conducting interviews, there is a problem to be aware of leading questions. If the interviewer, involuntary or not, ask questions that can be determined as biased, there will be a problem of receiving invalid data as a result, and can in the worst cases, help to goad the data towards a desired result that the researcher wants. In regards to the questions asked in this thesis, there is no certain way of confirming or denying that all the questions asked were unbiased or not, since there is only one researcher working on the thesis, with no external input. If there were a non-biased part that could either validate or invalidate the answers given from the interviewees, there would be potential for the results to change.

Chapter 6

Summary and further work

In this chapter, there will be a summary of the findings and further research that can be done to continue the improvement of introductory programming courses at a university level.

6.1 Summary

This thesis has explored the idea of improving introductory IT-courses at the university level for non-it major students. The preliminary research done has shown that there is a rapidly growing area for implementing IT-studies as a more central part of several science-based study programs and that there is a real need for students to learn and work in these environments. The researcher has performed a preliminary questionnaire study on early year students that have completed the introductory IT-course, TDT4110, at NTNU, where they were asked questions related to the course and how they have applied these skills in later courses. There has been done small interview sessions with students on how they experienced the course and how they have used these skills in other areas of their study program. These two data collections have been presented and discussed.

With the data gathering concluded and discussed in the last chapter, the following paragraphs will restate the research questions and shortly reflect on what has been done to answer them in the best way possible with the data collected.

- How can thematic exercises help improve students' performance in introductory programming courses?
- Can these tasks be, in some way, implemented into the existing system, and if so, can they replace the current exercises?

For the first research question, RQ1, there was a desire to look for any preliminary signs that students could benefit from reiterating on the current course. Where there any potential in checking to see if students would benefit from changing the contents of the exercise system to a more thematic focus. The results found that students were positive and receptive to the thought of changing parts of the exercise system to benefit them more, while also bringing in ideas for including other topics to the TDT4110 course, such as plots and graph generation.

The other research question, RQ2, is more varied with no clear conclusion. There are certainly many different ways that can be theorized for implementing such exercises, but all of these solutions are only mentioned as theorized ways, with pros and cons that need to be considered, though there is clearly a small chance of entirely replacing the existing exercise system with a new one that is specifically targeted towards niche scientific areas. These kinds of solutions are too time-consuming and a tremendous resource sink. Although this solution looks to be unlikely to succeed, there are many other theorized ways for implementing and improving students' learning of programming and how to apply these skills to their fields of study. See further work for more comments.

6.1.1 Limitations of the thesis

It must be acknowledged that the answers to the research questions are somewhat tentative and uncertain, due to some important limitations of the work. This section of the summary, states the limitations of this thesis, and clears out what it will not answer, as well as other shortcomings that have appeared throughout the thesis:

- The questionnaire has a very skewered participation, with a majority of responders hailing from the biochemistry. As such, there would be reasonable to assume that the questionnaire is somewhat biased from the biochemistry student point-of-view.

- There is no way to know if the participants in the questionnaire actually will complete their studies. If a participant later changes their study programs, it could also affect the results of the thesis.

- The questions about preferences between task topics (e.g., preferences between mathematics and a student's field of study) can be viewed as too abstract and may have confused students to answer differently than if the question was more clearly defined. If the student had been exposed to more examples throughout the semester, they would more likely have had a better understanding of the question and what it represented.

- The main limitation for the interviews is the limited amount of people interviewed. Five people are, if better than no one, still, a small pool of opinion to draw from, and with a larger pool of people to interview, one may have gotten a different opinion. It is worth noting since a majority of the interviewees had similar opinions and answers for several of the questions, there is a skewered result presented

6.2 Further work

As this is a fundamental and general study, there are several options to explore improving learning outcomes for students, but there has been mentioned several during the thesis. There are several options, both directly related to the thesis and its results, as well as other options that may have topics outside the scope of the thesis.

A more comprehensive survey

Since this thesis is a poorly synchronized towards receiving data during the TDT4110 course, it would be preferable to conduct the same or similar questionnaires during the fall semester, with new students. The chances will be that one would receive a higher amount of participants as well as a more distributed population among the study programs.

This suggestion could also nicely tie in interviews that explored more deeply the views that students have on programming in their scientific area, and how they utilize this. With more time, there could also be possible to construct and show examples of programming with relevant themes.

Compare thematic exercise program against existing exercise program

There is the fact that one should try to implement and run a demo-version of an exercise program during the semester on several students, to compare against students with the existing exercises, and compare either result on exams, or some other means, like tests throughout the semester. This would require that researchers prepare in advance, as well as closer cooperation with faculty to complete.

There are several factors, as mentioned previously, that needs to be addressed here, primarily the resource cost that this would happen upon faculty members, as well as the fact that teacher assistants may not have the required knowledge for the scientific topics.

Programming exercise generator

One other interesting concept is to investigate is to build a system that generates exercises for students, so that they can practice with new tasks. This exercise generator could also use themes from other study areas as a base for generating exercises. This was the original idea to be explored during this thesis but was later scrapped due to time constraints and scope.

It could also, with time, be expanded with the possibilities for the students themselves to write and submit their exercises, to diversify and increase the number of exercises available.

Bibliography

[1] Codewars: Train your coding skills. http://www.codewars.com.

[2] The Constitution of the Kingdom of Norway - - Lovdata. https://lovdata.no/dokument/NLE/lov/1814-05-17/ARTIKKEL_1#ARTIKKEL_1.

[3] Coursera — Online Courses & Credentials by Top Educators. Join for Free. https://www.coursera.org/.

[4] Faginformasjon - ITGK - NTNU Wiki. https://www.ntnu.no/wiki/display/itgk/Faginformasjon.

[5] Forskrift om rammeplan for ingeniørutdanning - Lovdata. https://lovdata.no/dokument/SF/forskrift/2011-02-03-107, .

[6] Forskrift om opptak til studier ved Norges teknisk-naturvitenskapelige universitet (NTNU) - Lovdata. https://lovdata.no/dokument/SF/forskrift/2016-08-25-1051#KAPITTEL_4, .

[7] Lov om behandling av personopplysninger (personopplysningsloven) - Lovdata. https://lovdata.no/dokument/NL/lov/2018-06-15-38/*#*.

[8] Kahoot! — Learning Games — Make Learning Awesome! https://kahoot.com/.

[9] Karakterstatistikk. https://sats.itea.ntnu.no/karstat//login.do.

[10] Khan Academy. http://nb.khanacademy.org.

[11] Codecademy, learn to code - for free. https://www.codecademy.com/.

[12] NSD - Norwegian Centre for Research Data. https://nsd.no/nsd/english/index.html.

[13] Office 365 - Wiki - innsida.ntnu.no. https://innsida.ntnu.no/wiki/-/wiki/Norsk/Office+365.

[14] Personopplysninger. https://www.datatilsynet.no/rettigheter-og-plikter/personopplysninger/.

[15] Spørsmål og svar: Ny lovgivning om personopplysninger - hva betyr det for forskning? http://www.etikkom.no/Aktuelt/gdpr-og-forskning/.

[16] UVa Course Catalog - Catalog of Courses for Chemistry (Unofficial, Lou's List). https://rabi.phys.virginia.edu/mySIS/CC2/Chemistry.html.

[17] Udacity - Free Online Classes & Nanodegrees — Udacity. https://www.udacity.com.

[18] Wolfram Mathematica: Modern Technical Computing. http://www.wolfram.com/mathematica/.

[19] Kontakt oss Kontaktpunkter UiO Adresse Universitetet i Oslo Boks 1072 Blindern 0316 Oslo Nødnummer Ved brann and Ulykker Og Alvorlige Hendelser Ring 22 85 66 66. Hovedside for IN1900 (+ MAT-IN1105, IN-KJM1900) - IN1900 - Høst 2018 - Universitetet i Oslo. https://www.uio.no/studier/emner/matnat/ifi/IN1900/h18/ressurser/hovedside.html.

[20] Peter Brusilovsky and Sergey Sosnovsky. Individualized Exercises for Self-Assessment of Programming Knowledge: An Evaluation of QuizPACK. 5 (2):22.

[21] Contact us Contact UiO Address University of Oslo P. O. Box 1072 Blindern 0316 Oslo Emergency In case of fires and Accidents or Serious Incidents +47 22 85 66 66. IN1900 – Introduction to Programming with Scientific Applications - University of Oslo. https://www.uio.no/studier/emner/matnat/ifi/IN1900/index-eng.html.

[22] Mihaly Csikszentmihalyi and Kevin Rathunde. The Development of the Person: An Experiential Perspective on the Ontogenesis of Psychological Complexity. In *Applications of Flow in Human Development and Education: The Collected Works of Mihaly Csikszentmihalyi*, pages 7–79. Springer Netherlands, Dordrecht, 2014. ISBN 978-94-017-9094-9. doi: 10.1007/978-94-017-9094-9_2.

[23] Christian Fossen. Chemical Engineering and Biotechnology - Masters Degree Programm - 5 years - Trondheim. https://www.ntnu.edu/studies/mtkj, .

[24] Christian Fossen. Mathematic institute - about. https://www.ntnu.no/imf/om, .

[25] IKT-Norge. Digital kompetanse og utdanning. https://www.ikt-norge.no/politiske-saker/digital-kompetanse-og-utdanning/, 2015.

[26] M.G. Kendall and A. Stuart. *The Advanced Theory of Statistics*. Number v. 2 in Griffin's statistical monographs and courses. Griffin, 1973. ISBN 9780852642153. URL `https://books.google.no/books?id=2igwuQEACAAJ`.

[27] B. Kitchenham and S Charters. Guidelines for performing systematic literature reviews in software engineering, 2007.

[28] Hans Petter Langtangen. *A Primer on Scientific Programming with Python*. Texts in Computational Science and Engineering. Springer-Verlag, Berlin Heidelberg, 5 edition, 2016. ISBN 978-3-662-49886-6.

[29] John McGrath, Jasmina Behan, European Commission, Social Affairs and Inclusion Directorate-General for Employment, and ICON-INSTITUT. *A Comparison of Shortage and Surplus Occupations Based on Analyses of Data from the European Public Employment Services and Labour Force Surveys: Labour Shortages and Surpluses 2017*. 2018. ISBN 978-92-79-81021-3. OCLC: 1039719819.

[30] S. Mitra, R. E. Lopez-Herrejon, D. Zimmaro, M. Johnson, and M. Schulman. An Assessment of the Effectiveness of Interactive Technology in an Introductory Programming Course for Non-Majors. In *Proceedings Frontiers in Education 35th Annual Conference*, pages S3C–S3C, October 2005. doi: 10.1109/FIE.2005.1612265.

[31] Briony J. Oates. *Researching Information Systems and Computing*. SAGE Publications, London ; Thousand Oaks, Calif, 2006. ISBN 978-1-4129-0223-6 978-1-4129-0224-3.

[32] Andrei Papancea, Jaime Spacco, and David Hovemeyer. An Open Platform for Managing Short Programming Exercises. In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, ICER '13, pages 47–52, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2243-0. doi: 10.1145/2493394.2493401.

[33] Lindsay B. Wheeler, Jennie L. Chiu, and Charles M. Grisham. Computational methods in general chemistry: Perceptions of programming, prior experience, and student outcomes. *Journal of College Science Teaching*, 45(3):83–91, Jan 2016. URL https://search.proquest.com/docview/1753971215?accountid=12870. Copyright - Copyright National Science Teachers Association Jan/Feb 2016; Document feature - Tables; ; Last updated - 2016-01-07; CODEN - JSCTBN.

[34] K. Yeh, Y. Xie, and F. Ke. Teaching computational thinking to non-computing majors using spreadsheet functions. In *2011 Frontiers in Education Conference (FIE)*, pages F3J–1–F3J–5, October 2011. doi: 10.1109/FIE.2011.6142980.

Appendix

6.3 visuals of results from questionnaire

The following pictures are taken from the results site of Microsoft Forms. These pictures are meant as additional visual aids towards the main findings for the questionnaire:

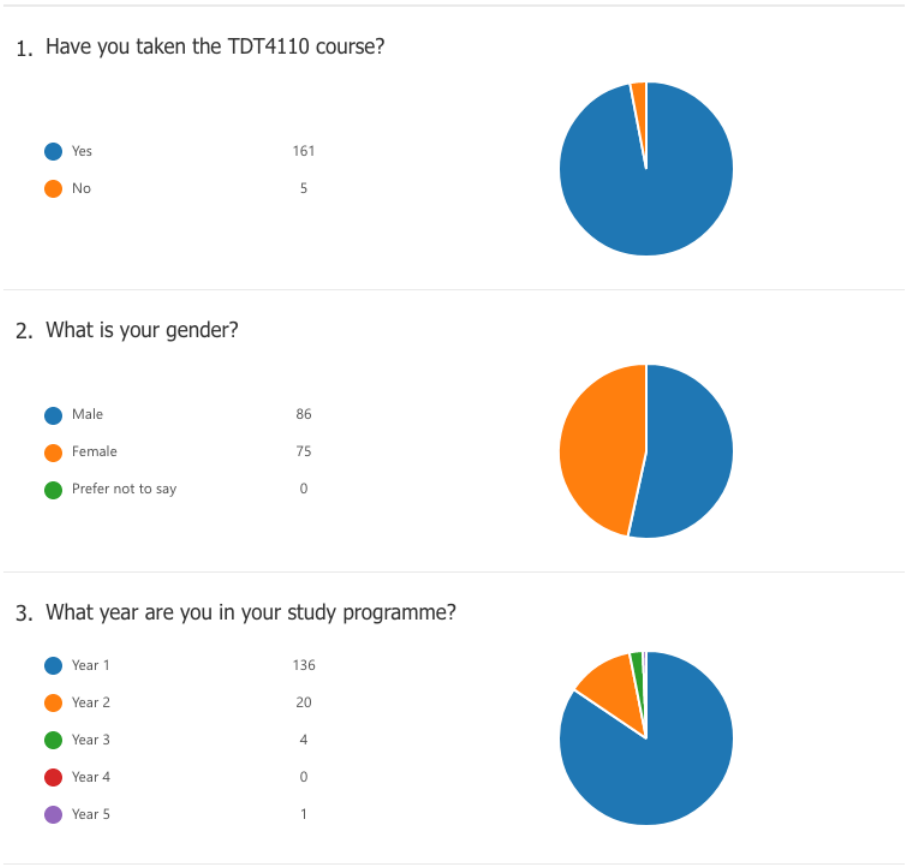


Figure 6.1: Result 1

4. What is your main study programme?

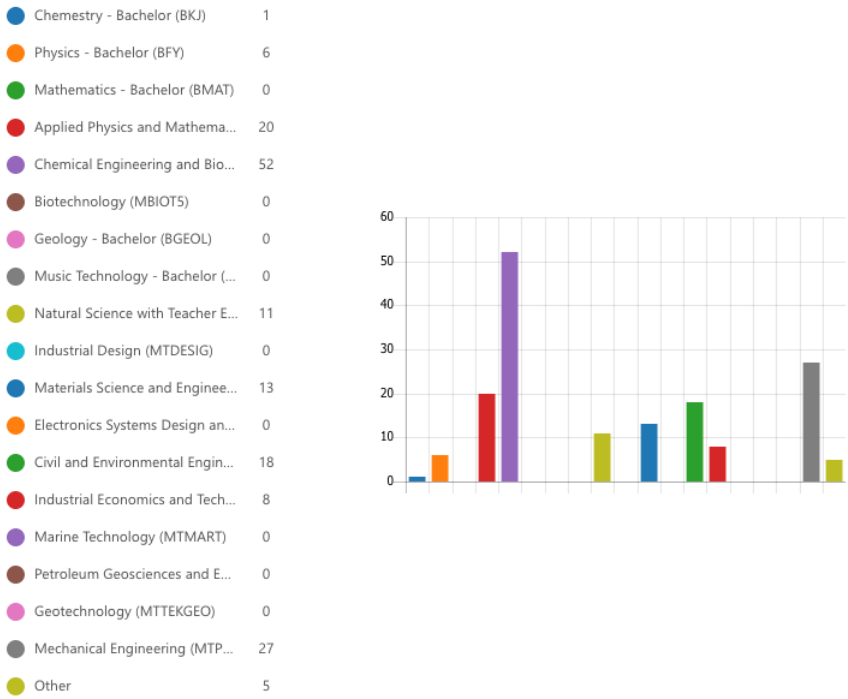


Figure 6.2: Result 2

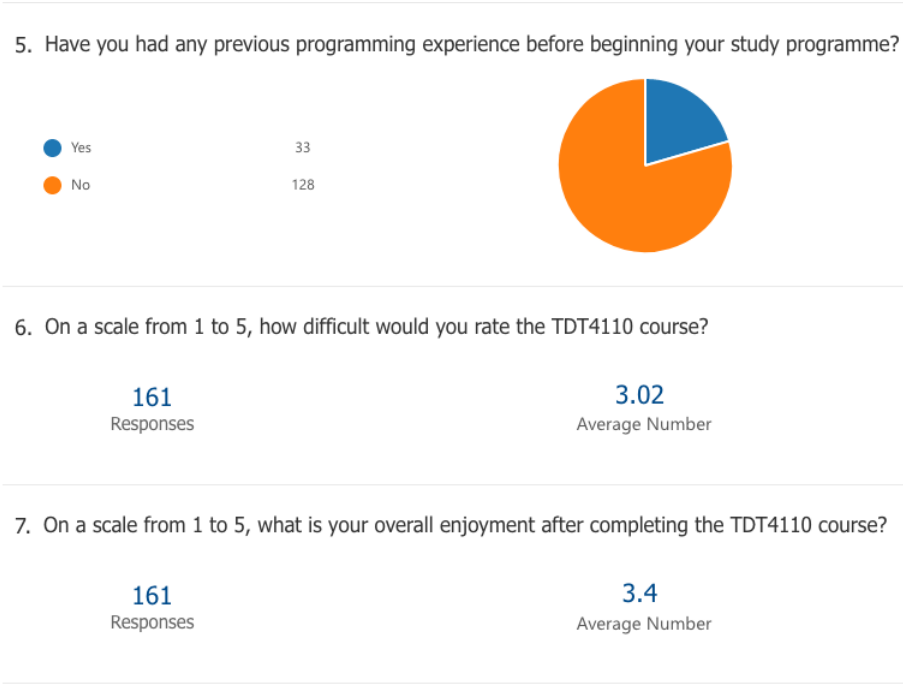


Figure 6.3: Result 3

8. The below questions are concerning your performance on individual parts of the course. On a scale from 1 to 5, how would you rate the difficulty of each aspect of the course:

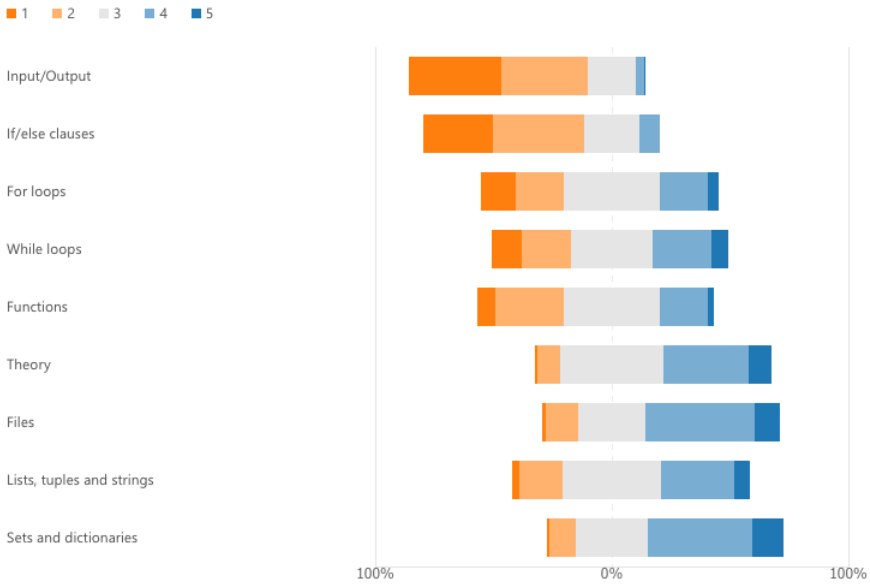


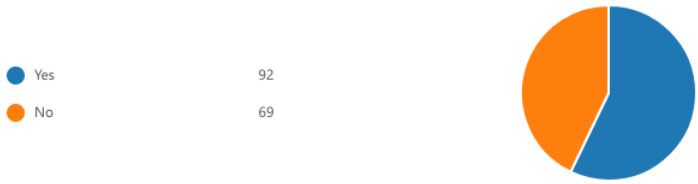
Figure 6.4: Result 4

9. On a scale from 1 to 5, how beneficial would you say programming experience will help you in your main discipline?

161
Responses

3.68
Average Number

10. Are you thinking about taking any more courses related to IT during your study programme?



11. On a scale from 1 to 5, how beneficial would you see the implementation of related course material from your own special field of study?

161
Responses

3.75
Average Number

Figure 6.5: Result 5

12. How is your preference between completely generic tasks and tasks related to your special field of study?

161	3.32
Responses	Average Number

13. How is your preference between completely generic tasks and tasks related to mathematics?

161	3.12
Responses	Average Number

14. How is your preference between tasks related to mathematics and tasks related to your special field of study?

161	3.36
Responses	Average Number

15. On a scale from 1 to 5, How willing would you be to participate in a demo with a few programming exercises tailored to your discipline?

161	2.55
Responses	Average Number

Figure 6.6: Result 6