UNIVERSITY OF TARTU
Institute of Computer Science
Software Engineering Curriculum

Mykhailo Dorokhov

Python programming module for non-specialized schools as an introduction to IT

Master's Thesis (30 ECTS)

Supervisors: Marina Lepp, PhD
Emanuele Bardone, PhD

Tartu 2024

Python programming module for non-specialized schools as an introduction to IT

Abstract:

This thesis describes the motivation behind, the emergence, and the subsequent evolution of the IT module, a set of three courses designed by the author, that has been running as an elective module in Tartu Annelinna Gümnaasium since 2020. The module combines Python programming language learning and project-based teamwork, serves as an introduction to the IT industry and provides career guidance for school students by presenting them the possibilities to study Computer Science at the University of Tartu. The aim of the thesis is to present a motivation that led to the creation of a new elective module in the school, describe how it's integrated into the state and the school's curriculum, break down course structure, explore the discovered teaching techniques, and reflect on the continuous improvements that the 4 iterations of the module have experienced. As a result, the best-suiting digital learning environment setup (Discord, Replit, Notion), a set of effective teaching techniques (pop-culture references, analogies, real-life-inspired homework and incremental learning) and events (guest lessons, "Chasing Unicorns" movie session, the University of Tartu Delta center visit and Pipedrive office visit with pizza party for the high-achievers), as well as a well-structured sequence of topics, has been established and described. The module's materials are shared via a public link for fellow teachers and researchers.

Keywords:

IT education, programming, teaching, K12

CERCS: P175 Informatics, systems theory; S270 Pedagogy and didactics

Pythoni programmeerimismoodul üldhariduskoolidele kui sissejuhatus informaatikasse

Lühikokkuvõte:

Käesolevas lõputöös on kirjeldatud autori kujundatud kolmest kursusest koosneva IT-mooduli motivatsiooni, teket ja sellele järgnevat arengut, mis on alates 2020. aastast läbi viidud Tartu Annelinna Gümnaasiumis valikaine moodulina. Moodul ühendab endas Pythoni programmeerimiskeele õppe ja projektipõhise meeskonnatöö, andes sissejuhatuse IT-tööstusesse läbi karjäärinõustamise, tutvustades neile võimalusi õppida Tartu Ülikoolis arvutiteadust. Lõputöö eesmärk on seletada lahti motivatsiooni, mis viis uue valikmooduli loomiseni koolis, kirjeldada selle lõimumist riigi ja kooli õppekavaga, tutvustada kursuse ülesehitust, uurida avastatud meeskonnatöö võtteid ning arutleda pidevate paranduste üle, mida mooduli nelja iteratsiooniga on kogutud. Selle tulemusena on paika pandud ja kirjeldatud kõige sobivama digiõppekeskkonna seadistus (Discord, Replit, Notion), efektiivsete õpetamise võtete kogum (pop-kultuuri viited, analoogid, päriselust inspireeritud kodutöö ja astmeline õpe) ning üritused (külalistunnid, "Ükssarviku" filmiseanss, Tartu Ülikooli Delta keskuse külastus ja Pipedrive kontorikülastus koos pitsapeoga edukatele õpilastele). Mooduli materjale jagatakse avaliku lingi kaudu kaasõpetajatele ja -teadlastele.

Võtmesõnad:

IT haridus, programmerimine, õpetamine, kool

CERCS: P175 Informaatika, süsteemiteooria; S270 Pedagoogika ja didaktika

Table of Contents

Introduction

In the extensively technology-driven world, Estonia not only follows global digitalization trends but, in fact, sets a standard as a country that has made a great leap towards a digital economy since the 1990s, becoming one of the role models in digitalization and e-government (Laar, 2008). Early digitalization focus paid off. In the public sector, innovations such as digital identity, digital signatures and online government services not only made citizens' lives easier but also played a significant role in manifesting a country's new global direction towards digitalization and openness to innovations. In the private sector, Estonia has experienced Skype, the country's first tech giant, and the subsequent "Skype effect": people involved in building Skype became founders or C-level executives in the next generation of Estonian startups, bringing their valuable experience of building truly big-scale and complex technical product (Saluveer & Truu, 2020).

Thus, the Estonian startup ecosystem has emerged, with all the necessary components, such as accelerators and incubators (Tartu Science Park, Tehnopol Incubator, TalTech Mektory, Prototron and others), conferences (sTARTUp Day, Latitude59, Lift99 and others), hackathons (Garage48 among others) and investors (Funderbeam, Contriber Ventures and others). Lately, a great addition to the ecosystem was UniTartu Ventures, an investment company that specializes in bringing university research and ideas to the global market, which is part of a global trend emerging in Estonian academic institutions (Mazina-Šinkar, 2022).

One important factor contributing to startup ecosystem well-being is access to talent (Velt, Torkkeli, & Saarenketo, 2018). A well-educated workforce is one of the keys to launching new businesses and ensuring their success, and in a situation where a startup hub or ecosystem is characterized by highly educated talents, not only does it contribute to its attractiveness and help to attract new investors but also further attracts new highly educated talent (Cohen, 2006).

Education is crucial in developing a country's talent pool. Schools and universities are where young people learn skills, knowledge, and creativity, enabling local businesses, government and non-government sectors to thrive. Estonian education is characterized as a high-performing system with equal possibilities for students from various socio-economic backgrounds to achieve results (Tire, 2021). According to the results of the Programme for International Student Assessment (PISA), in 2018 Estonian school students ranked first in reading and science and third in mathematics among all the OECD countries (OECD, 2019). This, however, was followed by the results slightly declining in PISA 2022, although still maintaining the position at the top of the leaderboard (OECD, 2023).

Reflecting the country's reputation as a leader in digital innovation and technology, there is a diverse range of possibilities to study IT (Information Technology) in Estonian universities. The University of Tartu, Tallinn University of Technology (TalTech), and Tallinn University all provide IT study programs at the undergraduate, graduate, and doctoral levels. At the bachelor's level, the classical Computer Science bachelor's program ("*Informaatika bakalaureuseõpe*" in Estonian) is available at all three universities.

Despite the numerous opportunities to study, the dropout rate is posing a great challenge for universities. Kori et al. (2015) have determined that among students of IT specialties at Estonian higher educational institutions, who studied in 2013/2014, the dropout rate during the first study year was 32.2%. Two factors particularly stood out: students' overall interest in the IT field and how well the curriculum met their expectations. Students with a low interest in the IT field and different expectations towards their future studies were more likely to drop out as soon as during their first year.

While some students who drop out actually do so because they start working in the industry (Polidano & Zakirova, 2011), and thus still contribute to the country's economy and startup ecosystem, a significant share of dropouts have simply had different expectations towards the IT industry, programming and software development (Kori, et al., 2015). In other words, they were not informed enough before making a decision to apply to a Computer Science bachelor's program. What could have helped them to prevent such a costly mistake? In Estonia, for about 56% of the students, the first contact with programming happens at the university, where difficult programming courses may cause stress and dropout (Kori, et al., 2015). What if they could have a chance to try programming before university, at school?

Lees (2006) writes that since Estonia regained independence, the state has supported IT education in schools in general. Early in the 1990s, Lennart Meri's "Tiger Leap" national initiative was created to modernize IT equipment and ensure internet connection in every school. This initiative was much needed yet mostly hardware-focused and aimed at allowing schools to have computer classes and develop teachers' and students' competencies so that various digital means can be used to support the learning process overall.

Programming, previously considered as a skill of the future, has now become a skill of the present. As a new form of literacy, much like reading or writing in the past, learning how to write code is not only beneficial because it helps students acquire highly demanded programming skills but also because it teaches computational thinking, improves problem-solving abilities, forces students to use abstract thinking and decomposition and overall nicely complements and intersects with the other Science-Technology-Engineering-Math (STEM) subjects (Wing, 2006). When it comes to programming, computer science or software development courses in Estonian schools, despite the country's clear digital focus, these courses remain elective (Vabariigi Valitsus, 2011). Thus, various schools do have elective IT courses with differing content, varying from actually learning programming basics to learning to use software like word processors, while some schools don't offer any elective IT courses at all, especially due to the fact that they struggle to find a teacher with the corresponding qualifications. And the struggle is real! A clear example is the numbers from the 2024 application campaign at Estonian universities: for the "Informatics teacher" Master's program, Tallinn University received only 0.75 applications per available place, making the program one of the university's 3 most unpopular study directions (Lumi, 2024).

This reveals that upper-secondary school is an important stage for introducing students to programming, making them try writing code with their own hands and think about possible solutions and practice teamwork, yet the lack of teachers who are qualified and motivated to teach IT courses in school continues to be a limiting factor. A possible solution to this issue might be participation in the "global race for talent" and attracting teachers from third countries, who would have the necessary knowledge, experience and motivation. While Estonia makes certain efforts to attract immigration of highly skilled workers and talents, these efforts lack a comprehensive and systematic approach and instead form a set of independently carried policies and scarcely coordinated measures (Ortega, 2014). Among the main limiting factors is Estonian language skills knowledge that the Estonian labor market requires (Kallas, et al., 2014), thus making foreign talent able to succeed mainly in industries where knowledge of the Estonian language is not necessary, the IT industry being a bright example with 25% of IT employees in Estonia being foreign specialists, as pointed out by former Prime Minister Jüri Ratas (Januzi, 2020). Since Estonia actually succeeds in attracting IT talent, the author being an example himself, what if a solution to the IT teacher shortage might be closer than one realizes?

This thesis presents a unique case of industry-education collaboration, where a foreign IT specialist successfully enabled the possibility to study IT-related courses for students of Tartu Annelinna Gümnaasium, a language immersion school that specializes in working with students from non-Estonian backgrounds and families. The aim of the thesis is to describe the motivation behind, the emergence, and the subsequent evolution of the IT module, a set of 3 custom-written courses that combine Python programming language learning and project-based teamwork, serving as an introduction to the IT industry and providing career guidance for school students by presenting them the possibilities to study Computer Science at the University of Tartu, or entering IT-related programs in the TalTech Tartu college or VOCO. The module is adapted to a school that already invests a significant number of hours into teaching students the Estonian language (Honcharova, 2019), providing a lightweight yet comprehensive introduction not only to programming in Python but also giving an overview of the IT industry in Estonia and globally.

Chapter 1 defines how the module is positioned within the state's curriculum, the school's specifics it accounts for, and how the module acts as a bridge between upper-secondary school and university for interested students. Chapter 2 describes the IT module, its final design choices, current structure, composition and digital learning environment setup. The chapter also contains an empirically established set of teaching techniques and activities. Chapter 3 uses AR (Action Research) methodology to reflect on the module's evolution: four incremental iterations of continuous improvements that the module has experienced since being introduced in the 2020/2021 study year. Chapter 4 analyses student feedback collected from the recent module's graduates to reflect on the latest module's iteration using the LOES-S model. Chapter 5 briefly depicts the future developments of the case, describing the dream of a possible extended version of the IT module with the focus shifted to product-first thinking. Finally, the last chapter is the conclusion of the thesis.

The thesis is written in the context of the Estonian educational system; thus the Estonian translations will be from time to time referenced in the parenthesis, like this ("*nagu nii*" in Estonian). To differentiate between school and university students, "school students" will be used for those in school and "university students" for those in university. If the context is clear, the term "student" will be used.

1 Background

This chapter describes the background behind the module's design choices. It describes the alignment with the state curriculum, accounting for the specific context of a school that puts effort into language immersion and has no STEM focus, and how the module builds a bridge from upper-secondary school to first-year university IT studies.

1.1 Alignment with the State Curricula

Estonian state curricula, in general, follow a decentralized approach (Lees, 2016). Estonian teachers have the freedom to choose their teaching materials and how they evaluate school students. The national curriculum focuses on what school students should achieve by the end of each school stage and gives advice on creating a school-student-centered learning environment. Schools are also allowed to create their own curricula, considering school students' interests and local cultural differences (Lees, 2016). Multiple lessons on the same subject with clear outcomes and structure are grouped using a notion of courses, with one course consisting of 35 academic hours. Different schools also use the notion of modules, one module usually having from 3 to 5 courses, to have a higher abstraction layer for bigger subjects or topics.

Informatics is an elective study direction, present in both basic school ("*põhikool*" in Estonian) and upper-secondary school ("*gümnaasium*" in Estonian) state curricula. In this thesis, the focus will be on what the state recommends for Estonian upper-secondary schools to include and how the module developed by the author fits into the state curricula.

The informatics state curriculum for upper-secondary schools describes a total of 11 courses. One can divide them into two main groups:

1. Six elective courses centered around the core Digital Solution Development Project ("*Digilahenduse arendusprojekt*" in Estonian). The set consists of the following courses: Programming, Software Development, User-Centered Design and Prototyping, Software Analysis and Testing, Digital Services and, finally, the Digital Solution Development Project itself.
2. Five more courses that serve as add-ons that schools may or may not include in their study programs, depending on their suitability to the school's curriculum. These courses are Cybersecurity, Computer Use in Research Projects, 3D Modeling, an Introduction to the Internet of Things (IoT), and Digital Footprint.

Salum et al. (2021), while describing and analyzing how the most recent edition of a new informatics curriculum was designed, state that the six core courses, centered around the Digital Solution Development Project, are created to better reflect the diversity of the IT sector and follow a classic software development lifecycle from identifying a specific problem and target group to delivering a solution within a team of 4 to 6 members with diverse backgrounds. This set of elective courses is ideally recommended for the 11th grade and unfolds creativity and cooperation skills between students who have taken different bits and pieces from the courses' selection, be it analysis, prototyping, design, programming or testing. The project's methodology aligns with modern educational principles, engaging students in a self-directed, entrepreneurial, creative, and cooperative manner. One important detail is that school students have the possibility of presenting their project as a research project ("*uurimistöö*" in Estonian) for their high school graduation requirement, further integrating informatics with other subjects and developing cross-curricular ties.

Here is a closer look at the courses of the Digital Solution Development Project:

1. The **Programming course** focuses on teaching students programming elements in a specific language, analyzing problems, creating algorithms and working programs, testing and debugging programs, and analyzing given program code. Topics include data types, variables, logical expressions, loops, strings, arrays, functions and data exchange. The course combines theoretical knowledge with practical programming tasks, emphasizing collaborative and self-directed learning methods.
2. The **Software Development course** teaches students to understand software development stages, analyze and solve problems using algorithms, create prototypes, and manage version control. It covers application development stages, data processing, and advanced programming constructs like double loops and recursion. The course also emphasizes collaborative learning and practical programming tasks.
3. The **User-Centered Design and Prototyping course** helps students to understand the design process, conduct user-inclusive design sessions, create paper and interactive prototypes, validate prototypes with users, and evaluate user experiences. It includes competitor analysis, requirement elicitation, user personas, and designing prototypes. The course focuses on creative practical tasks, collaborative work, and ongoing feedback from the teacher.
4. The **Software Analysis and Testing course** explains how to use correct terminology, analyze existing software, conduct requirements analysis, plan testing processes, create test cases, and document and present testing results. Topics include software quality, types and stages of analysis and testing, functional and non-functional requirements, test data and documentation. The course involves analyzing and testing real-life case studies in a step-by-step process.
5. The **Digital Services course** makes students understand various digital services, describe their functionality, analyze security risks, and compare different startup life cycles. It covers e-state and e-services, startup companies, basic ICT principles, service management cycle, and digital service regulations. The course combines theory with practical assignments.
6. Finally, the **Digital Solution Development Project course** itself is where students describe their future product target groups and their needs, analyze existing solutions, understand project risks, create documentation and present project results. It includes forming project teams, creating project plans, defining requirements, executing the project, and managing tasks. Students apply theoretical project management techniques in practical activities with continuous support from instructors and (ideally) industry experts.

All-in-all, these 6 courses provide a great way to experience the software development lifecycle in a group work environment. The curriculum states, that the main goal of studying informatics direction in upper-secondary school is for school students to gain a deeper understanding of the field so that interested school students would acquire the necessary basic knowledge and skills that would create opportunities for them to plan their future careers in IT industry, as well as for students interested on other specialties to indirectly use and implement IT in the respective fields (Vabariigi Valitsus, 2011).

According to the state curricula, the student study load in the upper-secondary school is 96 courses. Of these, 63 are compulsory, and in schools where teaching is partly in another language, this number is 72. This leaves schools with 33 (or 24 respectively) courses to be filled with electives. This leaves plenty of room for the Digital Solution Development Project courses, provided a school has a qualified and motivated teacher to deliver business analysis, prototyping, programming, as well as product and project management materials.

1.2 Alignment with Tartu Annelinna Gümnaasium

Tartu Annelinna Gümnaasium is an upper-secondary school in Tartu. It is situated in the iconic constructivist district of Annelinn. Tartu is the second-largest city in Estonia and the "student capital" of the country. The country's most prestigious university, the University of Tartu, is located in the town.

The school specializes in working with students from non-Estonian backgrounds and families. This used to include mostly local Russian-speaking kids; however, since the beginning of the full-scale Russian invasion of Ukraine, the school has welcomed more than a hundred kids from war refugee families (Olesk, 2022), making Tartu Annelinna Gümnaasium truly an international school.

The school utilizes a language immersion model to help students acquire much-needed Estonian language skills. This system emphasizes acquiring the target language not only via language lessons, but with the majority of other subjects being taught using the target language as a language of instructions (Mehisto & Asser, 2007). This proves to bring results, as most of the students (82.4% in 2022, 87.1% in 2021, 86.6% in 2020) successfully pass the Estonian language exam, earning the B2 level necessary to continue education in Estonian-language-taught university programs (EIS, 2024). In order to reach these impressive results, the school invests a significant amount of study hours into teaching students the Estonian language and languages overall. As of 2019, Tartu Annelinna Gümnaasium is investing 1120 academic hours (32 courses) at the upper-secondary school level in learning languages, whereas an Estonian-medium school of comparable prestige level and geographical location invests only 910 academic hours (26 courses) at the upper-secondary school level (Honcharova, 2019).

This investment is truly significant: from the 96 compulsory courses as required by the upper-secondary school state curriculum, 32 courses are dedicated to language learning in Tartu Annelinna Gümnaasium; thus, language learning takes exactly one-third of the whole upper-secondary school study load. The inevitable consequence of this is that while having a strong plan with investing hours into learning languages, the school has less room for elective courses, especially highly specific ones like the programming course.

The school offers students the possibility of choosing elective study modules to complete the required elective courses. A module is a set of 3 courses, totaling 105 hours (Tartu Annelinna Gümnaasium, 2022). Thus, a 105 academic hours module is a great scope for a possible lightweight yet all-in-one set of IT courses for students to try programming hands-on and get an idea of what the IT industry is.

1.3 Alignment with the University of Tartu

Estonian universities benefit from motivated students who have tried programming before at the school level as they're less likely to drop out from IT-related bachelor programs and perform better during their studies (Kori, et al., 2015). The University of Tartu, established in 1632 by King Gustavus Adolphus of Sweden, is one of the oldest universities in Northern Europe. For school students who graduate from school in Tartu, continuing to study at the University of Tartu is a natural way of progressing their studies. Thus, the possible module must take into consideration the specifics of the university and play the role of a bridge between school studies and entering the university for motivated students.

Python has a worldwide reputation as an excellent choice to introduce the fundamental ideas of programming (Stajano, 2000). The University of Tartu has been introducing programming to first-year computer science students via Python for already more than a decade,

from 2009 (Leping, et al., 2009). Python is a programming language used in the university's most successful programming "Fundamentals of Programming" MOOC (Lepp, et al., 2017) and is also a language used during the entrance exam to the university. Thus, it makes Python a natural choice for a programming language for a possible IT module that would teach basic programming concepts and serve as an introduction to IT industry as a field.

1.4 Similar Initiatives

There are existing initiatives in Estonia that are aimed at enabling schools to provide IT-related elective courses to upper-secondary school students.

DigiTaru (https://web.htk.tlu.ee/digitaru/programmeerimine/) is a digital coursebook containing materials for a "Programming" elective course for upper-secondary schools. Its materials are based on the "Introduction to Programming" MOOC, developed by the Institute of Computer Science of the University of Tartu. This initiative eases the burden on teachers by eliminating the need to search for or create content, allowing them to focus more on teaching. Still, it is expected that the school has a qualified teaching staff to support students during classroom learning.

The **Basic Programming Course** (https://taltech.ee/koostoo-koolidega/valikained) from TalTech is a self-study, free online course aimed at upper-secondary school students with little to no previous exposure to programming. It allows students to have a programming class without depending on whether their school has one. However, the self-study nature of the course poses a limiting factor for school students with poor organizational skills.

"From Technology Consumer to Creator" (https://didaktika.cs.ut.ee/progttl/) is an online course with a support remote mentor, organized by the Tartu University Informatics Didactics Working Group. The course content is developed in alignment with the state curricula's 'Programming' course, enabling school students whose schools are unable to offer a programming course on-site to attend one. Similarly to the previous initiative, school students with poor organisational skills may struggle due to remote learning nature of the course.

Codesters Club (https://www.codesters.club/) is a private-backed program supported by the Riesenkampff Foundation and local Estonian IT companies. Codesters mission is to provide modern "Digital Product Development" curriculum to schools, free of charge. Program supports schools with mentors and study materials. A unique feature of the curriculum is extreme product-first thinking, with courses ranging from business analysis, design, and prototyping to full-stack web application development. The full program runs throughout the whole upper-secondary school, from 10^{th} to 12^{th} grade, and requires the school to allocate 14 courses or 490 academic hours, which is not possible in every school.

Koit (https://koit.io/) is a young Estonian EdTech startup developing an integrated learning environment that combines study materials, a development environment and self-assessment. The team's goal is to create an engaging all-in-one online solution that makes self-studying programming possible. As of the writing of this thesis, the "JavaScript Basics" and "Python Basics" courses have been released, and the team is slowly building their way to a new-generation online courses platform.

To conclude, while various opportunities of different support levels exist, from full self-study to remote mentor support, they all seem to be a fix to the situation where a school does not have a qualified and motivated staff to offer a an on-site programming course.

2 Course Overview

This section describes the module's design, structure, and how it aligns with state curricula recommendations, simultaneously responding and adapting to the specifics of the Tartu Annelinna Gümnaasium as the target school and the University of Tartu as a most natural future educational step for school students in Tartu.

2.1 Design

Tartu Annelinna Gümnaasium already has a large number of courses dedicated to language learning and wants to keep the selection of elective courses broad, not only STEM-oriented. An ambitious goal was set to design a full-rounded introduction to IT as an industry and possible career in 105 hours, forming a three-course elective module.

Bird's eye view

The module is designed as an elective module for the 11th graders of Tartu Annelinna Gümnaasium. It consists of three courses, each 35 academic hours long, totaling 105 academic hours. Table 1 shows the main module features:

Table 1. Main features of the IT module

Feature	Description
Target grade	11
Module type	Elective module
Amount of academic hours	105 (3 courses, 35 hours each)
Grading	Non-differentiated (passed / not passed)
Link to module materials	https://it-module-tag.notion.site/
Language of materials	English, Russian
Language of instructions	English, intermediate Estonian, Ukrainian and Russian

The programming language used is Python, which is a language suitable for beginners, actively used in the IT industry, and the language that the University of Tartu students start with during their first semester. This way, the module's graduates will be better prepared should they decide to pursue an IT career at the university.

The course's content is written in English and Russian. This is partly due to the author's ongoing journey of learning Estonian, the insufficient level of English among the school students (Honcharova, 2019) for the course to be entirely in English, and finally due to the desire for language not to be a limiting factor when discovering possible IT talents. During the lesson, the author can provide support in English, Ukrainian, intermediate Estonian and Russian, according to his language skills.

The module consists of 30 topics (each topic takes 2 academic hours), 12 homework assignments and a final project. To get a pass, a student must complete 75% of the homework and pitch a final project.

Composition

The first two courses of the module, called "Programming Basics 1" and "Programming Basics 2", are focused on introducing school students to the main concepts of programming. The third course, called "IT Project", is a group work on a project, with elements of project and product management. In Table 2 one can observe the module's collaborative work dynamics and course breakdown. For a detailed list of lessons, see Appendix I.

The module increases the amount of collaborative work the school students do with each course. In the first course, all the homework is done individually. It is crucial for school students to work and think on their own when they're first exposed to programming (Grover & Pea, 2013).

Then, in the second course, the requirement is added to complete all the homework with a partner. Moreover, that person should be a different one for each new homework. Once a school student gets comfortable with writing simple code, it is beneficial not to stay in one's bubble but to observe and explore how different people tackle the same task and the way they approach and think about the problem (Xu, Wu, & Ouyang, 2023).

In the third course, school students work on a project in groups of four. One student takes the lead and plays the role of team leader. This course is designed to prepare school students for the collaborative nature of the tech industry.

The module ends with a final project pitch, where school students present their project in a startup manner: talk about the problem they're solving, explain the main features in simple language as if talking to future customers, yet explain the business side as if talking to potential investors. Each pitch concludes with a live demo, where students should create a narrative to walk the audience through the main features of their product. The teacher asks school students to prepare their final pitch in English. This is done to further force them to learn and use the vocabulary specific to IT industry, as well as get practice of public speaking in English. Also, the teacher invites guests from the industry, thus making it possible for every party to communicate during the final project pitch.

Table 2. The IT module structure

Course	Collaborative work	Key topics
Programming Basics 1 (35 hours)	Individual work	Variables; strings; data types; math; if statement; binary logic; validation; functions; loops (while);
Programming Basics 2 (35 hours)	Pair-programming (2 person)	Loops (for), data structures (lists, tuples, maps and dictionaries), modules and files; cryptography; algorithms; functional programming; object-oriented programming; API-requests; developing a Discord-bot or Telegram-bot;
IT Project (35 hours)	Teamwork (4 person)	Project management (decomposition, prioritization); high-level product development lifecycle (requirements, competitors analysis, MVP);

Drawing the parallels with the state curricula, both “Programming Basics 1” and “Programming Basics 2” resemble the “Programming course” from the state curricula, although at a slower pace, with “Programming Basics 2” having the majority of the elements from “Software Development course”. The final course, “IT Project”, follows the “Digital Solution Development Project” course, although the elements of competitors’ analysis and software development lifecycle are reduced to a very basic level to give students a general understanding and try only basic concepts.

On Table 3 one can see the module’s weekly organization:

Table 3. The module’s weekly schedule

Weekday	Class	Homework
Monday	-	Previous week homework’s deadline at 23:59
Tuesday	2 classes	Homework grades are revealed
Thursday	2 classes	Homework of the current week is introduced

Schedule-wise, lessons are held twice a week, in blocks of two (a total of four academic hours per week). It has been empirically figured out (more about that in Chapter 3) that Tuesdays and Thursdays are the best days for the module. One homework assignment is given on Thursday, with the deadline set to Monday 23:59 of the next week.

Study environment setup

The module’s setup uses the 3 digital tools: Discord, Notion and Replit.

Discord (https://discord.com/) is a synchronous communication platform, particularly popular among school students (Ceci, 2024). It is similar to the platforms used in the IT industry, such as Slack or Microsoft Teams; however, the choice of Discord benefits from the fact that most of the students already use the platform for gaming or hanging out with friends (Fonseca Cacho, 2020). The main idea behind using Discord is so that students can stay connected to the teacher and each other to ask questions, organize group work and discuss assignments together, all in one place, at any time of the day. For each iteration of the module, a new Discord server is created.

The following channel structure is followed for the module’s discord server:

📝 ***announcements*** channel permissions configured that allow only the teacher to write posts. This is the channel that the teacher uses for announcements, news, and updates. This is also where the teacher posts statistics after each homework assignment is completed.

💬 ***chat*** channel is the channel where everyone can write messages. It is aimed at supporting study-related conversation, where school students can ask questions from the teacher, but also from each other. The “slow mode” is configured, allowing everyone to post only once per minute, which makes every message more valuable, and forces students to think twice before posting both questions and answers. This is to create a more formal atmosphere in the channel.

☕ ***random*** channel is where everyone can write messages and there is no “slow mode” configured. This is where school students can talk about topics not directly related to the module, enabling more informal communication.

Notion (https://www.notion.so/) is an all-in-one productivity software that allows to create, organize, and manage notes, tasks, and projects in a highly customizable workspace. It is a trending tool that provides a simple and clean user experience (Teja, 2024). In the context of a module, it is used as a knowledge management system, effectively serving as an online coursebook that has all the lesson content in a clean and organized way, allowing to use text formatting, syntax highlighting and audiovisual elements, like embedding YouTube videos, GIFs and tiktoks.

Figure 1 contains an example of how the module topics look like in Notion.

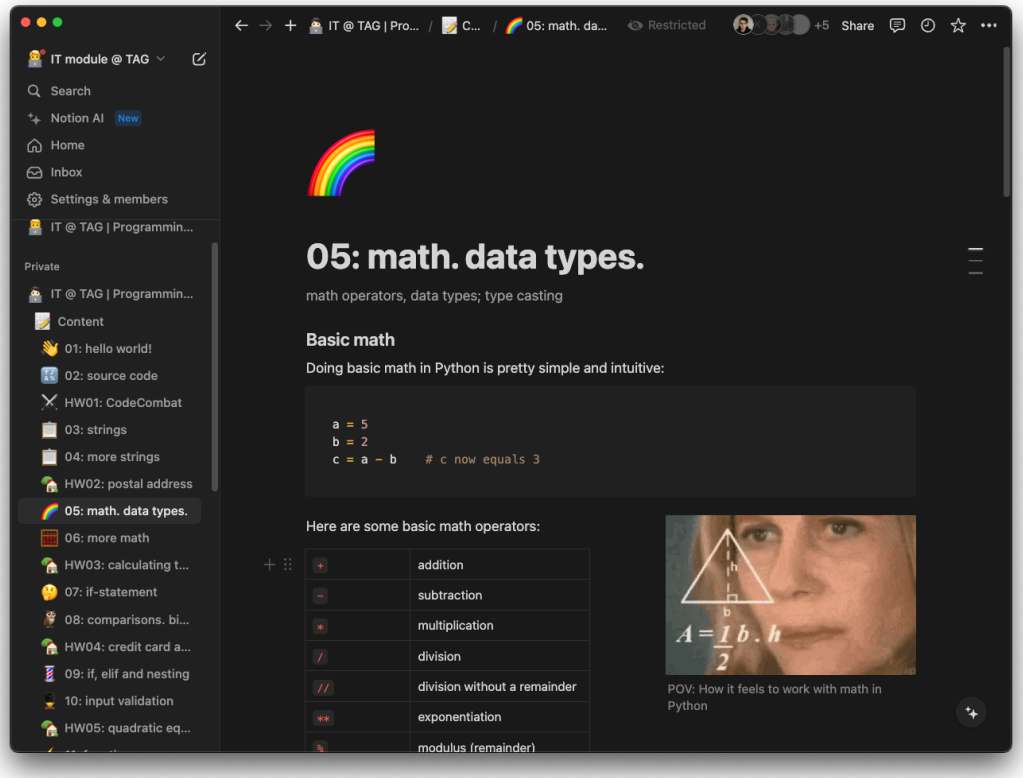


Figure 1. Example of the module's topic content in Notion

Replit (https://replit.com/) is an online coding platform that provides a collaborative environment for developers to write, test, and deploy code across multiple programming languages, including Python. It is designed to simplify the coding process with real-time collaboration features, integrated development tools, and instant hosting capabilities. Students can write code regardless of how powerful their laptop or desktop is; they don't have to install anything, and they can work with their code in a Google Docs style – by picking up the work at home where they previously left at and seamlessly collaborating with their peer during the pair-programming and group work.

A project in Replit is called "Repl", from the acronym "Read, Evaluate, Print, and Loop" that describes an interactive programming environment that allows users to enter expressions or commands, execute them, and see the results immediately. Students are asked to create a new Repl for each exercise and invite their friends in case the task has to be done in pairs or in groups.

On the Figure 2 one can see a Repl opened in Replit environment:

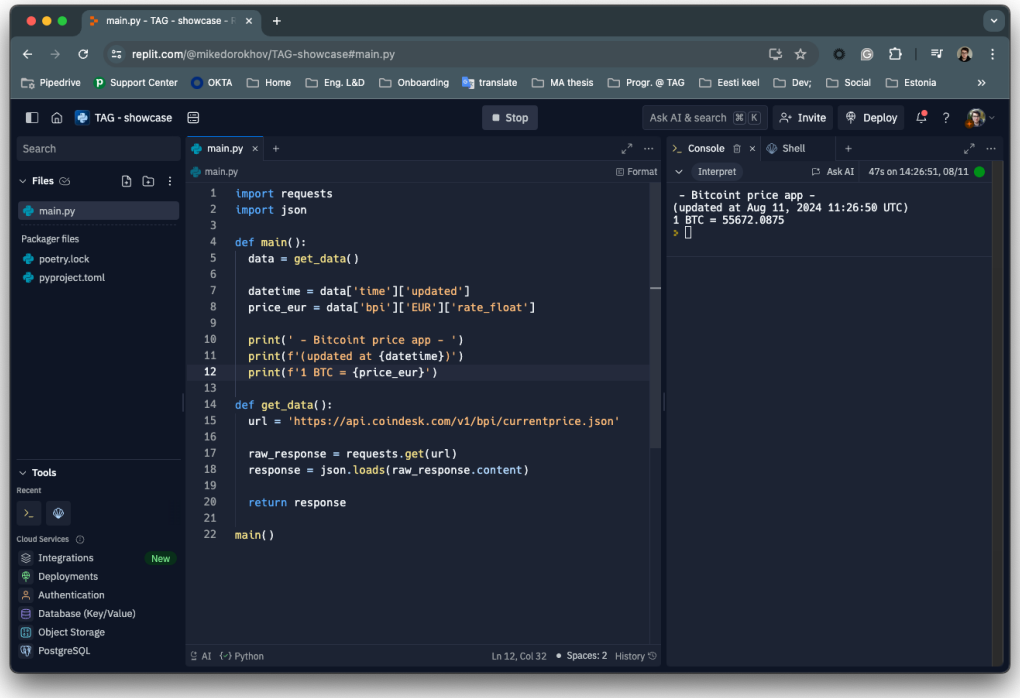


Figure 2. Replit – a development environment and platform

These three tools work in harmony to create a comprehensive learning environment, incorporating a communication platform, the development environment, and the digital coursebook.

Module's techniques

The author empirically came up with a set of multiple suitable techniques for use in class and when creating materials:

Pop culture references are similar situations, phrases or images from a famous movie, computer game, video or meme that fit into the context of a topic being learned and can be included in study materials or presented during classes. While this technique is already proven applicable for English language classes (Lin & Cheung, 2014); TikTok, YouTube videos and overall meme references serve multiple purposes during the module: capturing school students' attention and overcoming certain study fatigue moments during long classes, actually explaining the study materials, and making students understand that a teacher speaks the same language as they do.

Analogies are a way to explain complex concepts using real-life examples. In the same way as pop culture references, if done right, analogies can help explain study materials and increase engagement.

Real-life-inspired homework is a crucial part of the course. All the homework tasks throughout the course are carefully narrated to create a feeling that the application that the student writes solves a relevant, relatable and important problem.

Incremental learning is a technique where the previous tasks have to be refactored by students (see Appendix II). It increases solution understanding, code readability awareness and imitating real-world IT industry workflow (Izu & Mirolo, 2024).

Special activities

In addition to the ordinary lesson flow, special activities occur throughout the module:

Virtual guest lessons are rather short 20-minute video calls with Software Developers, ML Engineers and Engineering Managers the author invited using his connections. This gives a unique opportunity for students who are interested in the field and are considering entering the university to talk to a person already working in the field. Also, this further introduces students to the IT industry in Estonia, strengthening its image and improving students' understanding that there are a lot of great companies and projects taking place in Estonia, and one does not need to relocate elsewhere to find a great job or a challenge to tackle.

Watching the Estonian movie "Üksarvik" ("*Chasing Unicorns*" in English) usually happens close to the Christmas week. It's a great Estonian comedy movie about startups, teaching how to try, fail and try again. The movie overall sets the right mood before going to the Winter holidays.

Visit to the University of Tartu Delta Centre usually happens in the winter. It is aimed at softly introducing possible future study place to school students. In the later iterations of the module, the author invites the module graduates to talk to students about their experience as now university students, further connecting different generations of the same school. Seeing their schoolmates entering and succeeding in a university motivates school students and shows that everything is possible providing they put an effort.

Pizza and board games evening at the Pipedrive office is a rewarding activity for students who will do all the homework and successfully pitch the final project. After the module is concluded, and the grades are entered, the author invites the best students to Pipedrive office, on 16th floor of Paju 2 building, to have a pizza party, play board games as well as computer games together, chat and share the future plans.

2.2 Structure

This section describes the structure of the module's content, providing an overview of the weekly topics. See Appendix I for the list of lessons and homework assignments, and Appendix II for the list of tasks. The module's digital coursebook is shared under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license and is available via the following link: https://it-module-tag.notion.site/

Week 1. The module's intro, environment setup, and "tere maailm"

The module is overall designed to be as hands-on as possible; however, it starts with the first two lessons being an exception from this rule, where the teacher does a lecture (sort of an orientation of what to expect) that allows school students who are not sure about their elective module choice to carefully evaluate their decision. The school gives a 2-week grace period, for school students to withdraw from the course with no consequences; thus, these two weeks are designed to be enough to understand whether they want to continue with the course or not.

The intro lecture starts with the teacher introducing himself, where he is from and his educational, as well as professional path. Talking about his current job responsibilities at Pipedrive, as well as sharing what the company does and its history. It is important for students to discover that billion-dollar companies are created right here, in Estonia and by Estonian founders, to spark interest and inspire them that everything is possible.

Then, the disclaimer is made that the best students, who will do all the homework and successfully pitch the final project, will get a chance to join board games and pizza evening in

Pipedrive office, on 16th floor of Paju 2 building, following (almost as a proof) with pictures of their older schoolmates who participated in the previous editions of this evening. This is followed by an introduction to the IT industry, in the world as well as in Estonia. The teacher discusses what an IT company is and how its economics are different from those of an ordinary company. The teacher also discusses the Estonian IT landscape, unveiling its unique product-oriented focus and showcasing Estonian success stories, such as Pipedrive, Skype, Wise, Bolt, Glia, Veriff, Starship, Click&Grow and others.

Then, an important message is shared that the IT industry is not only about programming. When developing applications and services, besides software engineers, there are a lot of other roles involved in the process, such as product designers, product managers, recruiters and others. Actually, basic programming knowledge is useful far beyond the IT industry, for example, a production plant worker is considered really high-skilled when having the ability to work with machinery with CNC (computer numerical control); or a psychologist, who conducts massive scale surveys, benefits from the knowledge of Python or R to better analyze the data they gathered. This stresses on multi-discipline use of IT, which is also emphasized in the state curricula.

Then the narration shifts toward the latest developments in the IT industry, including self-driving cars, Machine Learning advancements and Large Language Models. The teacher gives examples of such researches happening in Estonia, providing an example of self-driving algorithms being in development jointly by the University of Tartu and Bolt (Sommer, 2020) or a self-driving bus by Auvetech (Kalda, Sell, & Soe, 2021), followed by the applications of computer vision in bioinformatics carried out by a startup called Better Medicine, that aimed to help with early cancer diagnostics using AI (Paulus, 2023).

The module overview closes the motivational part. The teacher explains that Python is chosen as a programming language to learn because it's simultaneously beginner-friendly, widely used in industry for web development and data science, and also the language used during the first year of bachelor studies at the University of Tartu. The teacher stresses that what one needs to join the course is not excellence in math or previous experience in programming, but rather a passion for solving complex challenges and an overall interest in a field. The course structure is introduced, and the criteria for getting a pass are explained. This follows with a study environment setup (Discord, Replit and Notion), and the first practical assignment, where students write their first "tere maailm" program.

The second day of the first week starts with a simple "introduce yourself" round. This makes students familiar with each other, as they are usually coming from different groups. The teacher leads the introduction round, giving freedom to a student in how they would like to introduce themselves, but asking every student in the end, "Why did you choose to join this module?" to have a picture of the motivations behind students coming to this class. The lesson continues with the teacher explaining what is a source code, variables, assignment operator, *input()* function and concatenation. Second-day lesson also uses a TikTok comedy sketch about an old man trying to order McDrive (Piltpull, 2021). The man of a respectable age struggles to keep up with the cashier, and it is used as an analogy and pop-culture reference, telling students that writing program code is the same as explaining to this same old man how to get to a bus station: one has to be very patient when giving direction, be verbose and don't miss any details, be clear and concise with step-by-step instructions.

The first week is concluded with homework that requires one to reach level 8 in the interactive coding game CodeCombat (https://codecombat.com/), which does a good job of making students practice simple programming concepts, such as command sequences and execution order in a playful manner.

Week 2. Strings

The first study day of the second week discusses strings. Throughout this lesson, students work with the email-app exercise, which starts as a simple email-building application, taking one's name and surname and formatting an email. However, it grows in complexity, making students use more concatenation and string methods and finally introducing them to string formatting.

The second study day expands the topic of strings, talking about *len()* operator, and introducing new string methods. Students write two small applications: positive-app, which makes any text sentence positive by replacing "not" with "definitely"; and nickname-app, which takes the student name and surname as input, and produces an American-style nickname, so that "Mykhailo Dorokhov" becomes "MD", with to trailing white spaces and with capital letter, even if typed otherwise by user.

The homework for this week is the Omniva-app. The legend says that Omniva reached out to help them automate postal address label formatting. The application should take a name, apartment number, house number, street, city and country as input and produce well-formatted postal address, ready to be printed and put on the envelope.

Week 3. Math and data types

The first study day introduces how one can do math in Python. At first, this seems quite intuitive and self-explanatory. School students are tackled with the how-old-are-you-app, which takes the user's birth year as input and outputs their current age. This is where things get interesting, as when trying to subtract from a variable received from user input, students get *TypeError: unsupported operand type(s) for -: 'str' and 'int.'*. This is when the teacher explains data types in Python. The analogy used here is that one can think of data as goods, and when transporting with a railroad, different kinds of goods need different cars. Food needs cars with refrigerators; oil needs tank cars, and people need passenger cars. Same way, different variables store different data of different types. After students learn typecasting, they successfully complete the exercise and are now ready for two more: currency converter and average grade calculator.

The second study day continues the math topic. Students learn different built-in math functions, such as *min()*, *max()*, *round()*, *abs()* and *pow()*, and then move to the *math* Python module. This also plays as a soft introduction to importing modules. The analogy used to explain a Python module is sticker packs in messengers: users usually have some emojis available by default. However, they can choose to import a sticker pack that will enhance their emoji selection. Students are challenged to upgrade the average grade calculator, now using rounding. Three more exercises that they perform are on Pythagorean theorem, on Logarithms and on circle area formula.

The homework for this week is the EMTA-app, or a tax calculator. The legend says that EMTA reached out to help them write a simple tax calculator. A user enters their gross salary, and the calculator shows a tax breakdown, displaying the user's net salary, the total cost for the employer, as well as the amount of income tax, social tax, unemployment insurance premium and pension fund contribution. Not only does this homework solve a real-life case but it also makes students learn about the Estonian tax system.

Week 4. Conditional operator

The first study day introduces students to if-statement and the comparison operator, stressing their deceiving similarity to the assignment operator. A real-life example used in the exercises is selver-app, where an application should output different prices, depending on

what the user answered about having a loyalty card. Students also learn about *elif* operator, and the lesson concludes with a close to real-life exercise to write a visit-estonia-app, where a user types an Estonian city name, and the application tells information about this city.

The second study day talks about comparison operators (==, !=, >, >=, <, <=) and logical operators (and, or, not). Here, the author cannot resist using the Marriage Logic Map meme, popular among programmers (DesPurpleLightning, 2020).

The homework for this week is the LHV-app, an application for LHV bank. The bank asked students to write an application that decides whether to grant or refuse a client a credit card. Students must implement the application flow using real-life conditions they can find on the LHV bank website.

Week 5. Conditions, nested conditions and validation

The first study day is dedicated to materials about *if*, *elif* and *else* operators, and when to use nested conditions versus a combination of conditions joined by logical operators. A nice real-life example used is an application, that calculates a fine for speeding in Estonia, as the fine depends on the range of a delta between the actual and allowed speed.

The second study day introduces students to input validation. Different string methods are observed, such as *isalpha()* or *isdigit(),* and how to use them to validate input. Previously done exercises, such as Omniva-app, are upgraded to feature input validation. Also, the teacher talks about guard statements, a code style that makes application code cleaner and more readable.

The homework for this week is an application that solves quadratic equations - it perfectly combines the usage of input validation, if-statements, math and string concatenation. Making these two weeks math-intensive helps to promote math knowledge among students, make them repeat certain aspects of school math and straighten interdisciplinary ties.

Week 6. Functions

The first study day is fully dedicated to learning the concept of a function in programming languages. From the author's experience, the function is a topic that introduces a sudden increase in the module's complexity. Thus, this concept is unfolded slowly. A good real-life analogy is a friend's birthday party. When you agree with friends to make a surprise for a birthday person, this is a function body, and when the day finally comes and you execute your surprise, it is a function call. The lesson ends with a single practice, composed of familiar tasks, now wrapped in the functions.

The second study day discusses refactoring and code organization. Students are instructed to revisit an application that calculates a circle area and refactor it, splitting it into two functions: *main()*, which handles input and output, and *calculare_area()*, which takes care exclusively of business logic, i.e. calculating a circle area knowing its radius.

The homework for this week is an application that calculates a sphere's volume.

Week 7. While loop

The first study day explains the basics of loops using a *while* loop as an example. The good old scene from the Simpsons is used as an example, where Bart has to write a certain line multiple times on a whiteboard. Students are told that a *while* loop would have made Bart's life easier.

The second study day continues the *while* loop topic, introducing two keywords: *break* and *continue*. The homework for this week, *guess-a-number* game, is also introducing during

the lesson – students start writing it during the class so that the teacher makes sure everyone understands the main game loop concept. During the homework, they are instructed to "release a DLC": add multiple features to the game, including a finite amount of tries, hinting to the player whether the entered number is bigger or smaller than the one they are trying to guess, and configure the game's settings. Next week, the teacher will use the game and the hints feature to introduce the students to the concept of binary search.

Week 8 is a consultation week.

Students can come to a class and ask the teacher to repeat and practice any previous topic. This is also a week when the "Chasing Unicorns" movie is being watched.

Week 9. For loop. Lists.

The first study day describes *for* loop, its similarities and differences to *while* loop.

The second study day introduces the first data structure – a list. A good example of explaining the difference between a variable and a list is comparing them to a single-family house and apartment block, respectively. Here some salt can be added about the fact that some of such "lists" can be seen right from the school window, obviously referencing Annelinn "paneelikad" and thus adding much-needed interactivity and fun to the lesson.

The homework for this week is about Instagram followers. Students are given two lists: a list with the user's followers, and a list of the people user is following. Students have to write an algorithm that will display accounts that the user is following, yet that do not follow a user back – obviously violating the main Instagram politeness rule.

Week 10. More data structures: tuple and dictionary.

The first study day introduces a tuple data structure, while the second study day talks about a dictionary. A great task for the dictionary is an emoji-converter application. The task has real-life implications, as all the modern messengers have the auto-replace feature for text-based emoji, turning emoji like this one ":)" to graphical, like this one: "🙂".

The homework for this week is nickname-app – a nickname database that contains data on a student's classmates' nicknames.

Week 11. Files. Cryptography.

The first study day focuses on working with files and different reading and writing modes. A real-life exercise from this lesson is parsing an imitation of a receipt from Prisma.

The second study day introduces basic cryptography concepts. The teacher explains how the Caesar cipher works, and then students have a chance to write a code that ciphers a text message. After this is done, the teacher takes on a challenge to decipher the messages of students by showing them the brute force technique, which is a great starter for a discussion on Caesar cipher's reliability. Then, the teacher talks about XOR-cipher, which works with binary data. As a challenge, students have to use the code provided by the teacher to decipher an image by trying out different keys. Students quickly learn that numbers that could serve as a key are from 0 to 256, which makes their task easier. The teacher concludes the lesson by talking about symmetric and asymmetric keys.

The homework for this week is to improve the previous homework, nickname-app, by making it save the data to a file, providing persistent storage. As a bonus task, this file might be ciphered to prevent unauthorized access to the data.

Week 12. Modules. Algorithms.

The first study day describes to students how to create their own module, with functions and logic. Students have a couple of tasks to refactor applications they previously made and to place functions into different modules.

The second study day touches important concept in computer science – algorithms. The teacher explains an algorithm that finds the smallest number in the list, and students have to write an algorithm that finds the biggest number in the list. Then, the next task is to write a code that outputs unique numbers from a list, and the advanced level task is to write a code that displays how many times every unique number is encountered. The last task combines both basic algorithms and data structure knowledge. The teacher touches on the topic of O-notation and recursion, showing how factorial calculating can be written both with and without recursion.

The homework for this week is to register on the Codewars website (https://www.codewars.com/) and to solve 3 different challenges on the platform.

Week 13. Functional programming. Classes.

The first study day talks about elements of functional programming present in Python, and in particular, functions *map(), filter()* and *reduce()*. An analogy to explain how *map()* works is photo editing on a smartphone: the teacher compares *map()* to applying the same filter to every photo in students phone's gallery. Analogy for *Filter()*is a bodyguard at a nightclub, who deals with evaluating every person in a queue to enter the club. The bodyguard does so sequentially, one person at a time, and decides to let a person into a club or not according to a certain set of conditions. The legendary tweet by Steven Luscher is used in study materials for further show how all three functions work (see Figure 3).

The second study day briefly touches on object-oriented programming elements in Python, and explains such concepts as class, object, inheritance, polymorphism and encapsulation. The teacher asks students' input on what entity to take as an example, and to create a *class* from. This improves the quality of discussions, such as whether a certain feature is a *method* or an *attribute*, and so on.

The homework for this week consists of a task to create a class for a social media post, create an instance of it and play with its functionality.

Figure 3. Map/filter/reduce in one tweet, by Steven Luscher

Week 14. API-requests. Telegram-bot

The first study day discusses how the Internet works and explains concepts such as client-server architecture, URLs, domains and IP addresses. Students are introduced to API-servers and JSON, and they are given a task to connect to a simple API-server and display the content it returns. On this day, the favorite task among students proved to be the task of using the Coindesk API server to retrieve the current Bitcoin price. When running the application with certain intervals, one can see almost in real time how the cryptocurrency fluctuates. Thus, the advanced level of this task is to make the application request the price in a loop, "remember" the previous price and show the current price trend – whether the current price went up or down compared to the previous indicator

On the second study day, students learn to create a simple Telegram bot, reusing the API calls code to embed it as one of the bot's commands. Starting from this week, there is no more homework.

Week 15. Multiuser support and data persistency.

During this week, students, together with the teacher, write a simple water-drinking tracking Telegram bot to showcase different advanced yet core features a chatbot should have. During the first study day, students make their Telegram bot have a multiuser support and implement persistent storage using Replit's built-in feature. On the second study day, they add complex logic to their bots so that it can support conversation, and learn how to invoke custom keyboard.

Week 16. Discord-bot and consultation

The first study day is dedicated to showing students how to create a Discord-bot. The similarity of the process highlights that all the chatbots are created in a way that has common steps and logic. This week is also used for a consultation class.

Week 17-24. IT-project

The last course, IT project, is introduced during week 17. Study organization changes in this course: lessons are no longer hands-on practice sessions but rather each team's standups, involving the teacher as a mentor for every team.

Students are told to form teams of 4 and come up with a couple of ideas to validate with the teacher. The teacher validates ideas to be relevant and technically accomplishable. The teacher introduces the student team to the basics of product management by explaining topics such as competitors' analysis and user persona and forcing student teams to decompose their ideas into requirements. Past this stage, the project management phase starts, where the teacher explains Kanban and how to further decompose requirements into tasks, prioritize them and parallelize work on the project by employing each team member's strengths.

After multiple development sprints, the teacher gives training on how to perform a good pitch. The pitch day is the final lesson of the whole module. The teacher pushes the pitch day to be fully in English and makes sure to invite someone from school management, the module graduates who are already studying at the university and someone from the industry to come as guests and ask questions. This makes the event feel significant and memorable for the module students. After the pitches are done, the teacher invites students to anonymously vote for the best product and the best pitch to hand over the prizes.

The last slide is shown with a link to the Discord server for the module graduates.

3 Iterations

This chapter describes the evolutionary changes and continuous improvements that the module experienced since being introduced to Tartu Annelinna Gümnaasium in the 2020/2021 study year.

3.1 Methodology

Action Research (AR) is an iterative research methodology that combines action and reflection to solve problems and improve practices within a specific context over time (Kemmis, McTaggart, & Nixon, 2014). The AR process typically consists of the following cycle: planning an intervention, taking action, observing the outcomes, and reflecting on the results.

The Figure 4 contains a high-level overview of the iterative AR cycle:

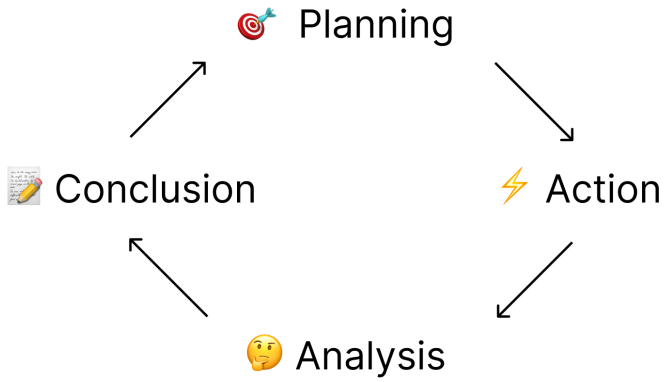


Figure 4. Action Research cycle

The module's iterations were analyzed mainly in two parts:

1. **Plan and act**, where the summer planning is followed by taking action of writing or updating the module's materials and conducting the lessons during the study year;
2. **Observe and reflect**, where the author uses a combination of techniques, such as observing every lesson and self-reflecting upon each lesson, as well as collecting verbal feedback in the form of short interviews from students after the module is completed and strictly after the final grades are released.

In total, 4 iterations are analyzed, from the 2020/2021 study year up until 2023/2024.

3.2 2020/2021

The first edition of the module was a pilot. The school agreed to experiment with a new elective module and observe whether there is a demand among students.

19 students joined the module, and 15 successfully completed all three courses, achieving a completion rate of 78.9%

Plan and act

The school's requirements were an elective module, three courses (105 academic hours in total), and non-differentiated grading. The author decided to follow the above-described structure, with the first two courses dedicated to hands-on learning and the last course being a group project. Lessons were held on Tuesdays and Thursdays at the end of the study day, which is a traditional time on the study schedule for elective courses. Homework was released every Thursday, with a deadline set for the following Monday evening.

Slack was chosen as a communication environment. A Slack channel contained all the lesson content and played the role of a digital coursebook. The author viewed it as a chance to introduce asynchronous communications, which are widely used in the industry. Replit.com was chosen as a zero-setup online coding environment to eliminate any setup friction.

In the third course, the author experimented with having students pitch their ideas and make others join the ideas they liked to imitate hackathons. In parallel, virtual guests from the industry were invited for students to ask career-related questions.

Observe and reflect

The module was overall perceived positively by students and, thus, by the school administration, leading to its inclusion in the school's selection of elective modules available for 11th graders.

At the end of the module, after the final grades were released, the author conducted interviews with students for them to express their feedback on the module. Students liked the hands-on nature of the module, working on a project in a group, and the author noticed how homework that included certain narration (for example, a made-up story about how one Estonian bank asked a class to help and write an application for them) got more engagement.

The main negative feedback was caused by the idea pitch for forming student groups. Majority of the students already had a friend group that they wanted to keep for the project.

3.3 2021/2022

The second iteration proved that there was demand from students, and the school decided to include the module in the permanent elective module's selection. This time, 26 students joined the module, with 18 of them successfully completing all three courses, resulting in a completion rate of 69.2%

Plan and act

During the second iteration, mainly due to a push from the school's administration side, an experiment was carried out by changing the study days for the module to Monday and Wednesday. Following the feedback on the narration-rich homework, those tasks that did not feature a story were rewritten, and now all the homework tasks have a realistic story behind them.

Since students liked working in a group, another experiment was to increase the amount of collaborative work by doing the homework tasks of the second course via pair programming, better preparing them for the project part. This way, the module would have a progressive transition from working on their own, pair-programming, and finally working in a group of 4 on a final project. Following the feedback on the idea pitch, it was turned into a team pitch, when students formed teams themselves and talked about their ideas for the teacher to validate their business and technical feasibility. The teacher acted in case there were students left behind and not included in the existing groups, joining them together.

Observe and reflect

During the traditional end-of-the-module feedback session, school students shared the following: positive feedback was received for the pair programming while doing homework, narration-rich homework got praised and as expected, school students very much liked the fact that they were able to keep their friend groups for the project. Surprisingly, friend groups also performed better, showing the importance of a good team bond in practice.

Negative feedback was received for the study day schema. The students struggled during the last two lessons on Monday due to having such a long day at the very beginning of the study week. Also, the first lesson on Monday meant that the homework had to be done by Sunday 23:59, effectively forcing students to work on it during the weekend, which, while inevitable due to students nature (Yang, et al., 2020), the author dislikes the very idea of. Also, Slack, as a communication and content environment, got slight negative feedback from students.

3.4 2022/2023

During the third iteration, the module was already a well-established part of the school study process. Gaining popularity among students, the module was ready for further improvements. 25 students joined the module, and 17 successfully completed all three courses, achieving a completion rate of 68.0%

Plan and act

Considering all the previous feedback, the decision was made to return to the Tuesday-Thursday study days setup and never change it since. The biggest change that year was a learning environment revamp: the author decided to address the concerns about Slack and switch to a more sophisticated yet expected-to-be pleasant-to-use setup: Notion and Discord. Discord is an excellent communication tool, and not only is it similar to Slack, but it is also popular among the target audience of the module (Ceci, 2024); thus, most of the school students didn't even have to create an account, and the introduction of the communication platform was seamless.

One new activity was introduced during this year, namely the visit to the University of Tartu Delta Centre. It was inspired by the fact that one module's graduate of the 2020/2021 study year successfully entered the University of Tartu and started studying Computer Science. The idea was to establish a tradition of current module students exploring the university and meeting with the module's graduates studying there as an example of what is possible after one completes the module and to create networking opportunities between school students and university students who have finished the same school.

Observe and reflect

The traditional end-of-module feedback session revealed impressive positive feedback. Study days moved back to Tuesdays and Wednesdays, the learning environment upgrade and a visit to the University of Tartu Delta Centre were perceived positively by the students. It felt that a course was finally taking its final shape, with the high-level features established.

Reflecting on the low-level details, the author felt that the functions, as a topic, were introduced rather late. According to the feedback, previously requested from the university, some of the topics were still missing, such as object-oriented programming, nested loops and recursion.

3.5 2023/2024

The fourth iteration of the module was showing signs of maturity. All the major elements were left unchanged, and only the details were tweaked. This year, 26 students joined the module, and 20 of them successfully completed all three courses, marking a completion rate of 76.9%

Plan and act

Further tweaking the content, the author moved the topic of functions closer to the beginning of the module. Functional programming elements (namely *map()*, *filter()*, and *reduce()* functions), object-oriented programming basics, nested loops and recursion topics were added, as well as more complicated topics regarding chatbot development, such as multi-user support and persistent memory storage.

As it was the fourth year of the module, there were already multiple students who had finished the module and were studying in the first and second years of the university. Expanding on the idea of connecting the different student generations, an alumni Discord community was created for students to keep in touch, share news and experiences, and for the module graduates to help school students choose and apply for university programs.

Observe and reflect

After completing the module, students highlighted in the interviews that the addition of functional programming elements, object-oriented programming basics, as well as multi-user support and persistent memory storage topics for the chatbots, were both interesting and important additions. A couple of students were actually happy that the chatbot that they developed is now a "serious application", and expressed a wish to continue developing their "business".

The students' annual verbal feedback also marked the alumni community created on Discord as the biggest achievement of the fourth iteration of the module. It gave the module a bigger context, not only as a place for students to try programming hands-on to help them decide whether it's something for them or not but also as a place with its own unique identity, a club of school students and module graduates interested in the IT industry.

4 Feedback

This chapter analyses student feedback collected from the recent module graduates to reflect on the latest iteration of the module in the 2023/2024 study year. The feedback collection methodology is explained, and the feedback results are analyzed.

4.1 Methodology

After the module's four iterations of continuous improvements and fine-tuning, it was necessary to understand how students would assess the state of the module and what further improvements and changes were expected. To avoid any pressure, the idea was to create an anonymous questionnaire distributed to students after their grades were released. Such a questionnaire would simultaneously assess the quality of the content and study environment setup and also give insights into the quality of the teaching.

To combine these two goals, two sources of questions were chosen: the LOES-S (Learning Object Evaluation Scale for Students) questions to assess the quality and engagement of the module's content and environment (Kay, 2009) and questions from the SIS (Study Information System of the University of Tartu) questions to assess the quality of teaching.

The LOES-S is a questionnaire specifically tailored for the school environment that is aimed at evaluating learning, quality, instructional design, and engagement aspects of a learning object - an interactive web-based tools that support the learning of specific concepts by enhancing, amplifying, and optionally guiding the cognitive processes of learners (Kay, 2009). In the context of this thesis, the learning environment setup (Notion as a module's digital coursebook, Replit as a web-IDE, Discord as a communication platform) will be considered as a learning object.

The SIS course feedback is used for every course taught at the University of Tartu. In the questionnaire, questions that cover teaching style, methods, and additional impact on students' career choices are selected.

Thus, the questionnaire combines research-proven questions from LOES-S and rather empirically-proven questions from SIS. Including questions from SIS makes the module feedback also comparable to university courses' feedback.

4.2 Questionnaire

The questionnaire consists of four blocks: learning, quality, engagement and teaching. The first section focuses on the learning aspect of the module's learning environment setup; the second section focuses on the quality aspects of the module's learning environment setup, while the third section contains questions that reveal engagement in the module overall. Finally, the fourth section contains questions about teaching quality, an impact on school students' career choices, and the overall perception of the module being valuable for a student. The first three blocks were based on LOES-S questions, while the last block was based on SIS questions.

Where applicable, the phrase "Learning Object" in the original LOES-S questions is substituted with "the module," "the module content," or "the learning setup (Notion, Replit and Discord)" to help school students better understand the questions. All questions have also been translated into Russian, and the translation is presented under the English version of the question. The questionnaire is available in Appendix III.

The questionnaire is created in the TypeForm (https://www.typeform.com/) environment, being rather a design-over-functionality choice to engage students in completing the questionnaire. Figure 5 shows the questionnaire welcome page and a question page.

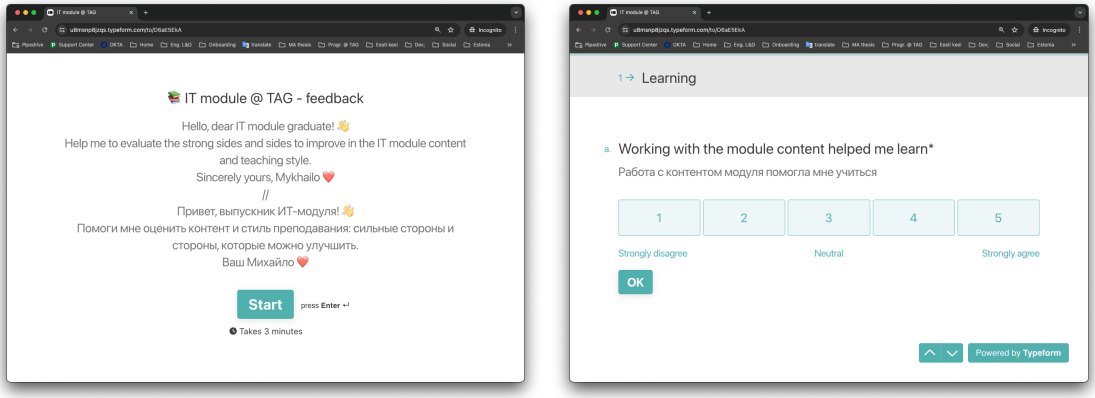


Figure 5. Questionnaire in TypeForm

4.3 Results

The questionnaire was started by 23 students (out of a total of 26 who participated in the 2023/2024 module) and completed by 20 students. The completion rate was 86.9% of those who started, and 76.9% of those who have registered to take the module. On average, it took students around 3 minutes to complete the questionnaire. The Likert scale was used for most of the questions, except the two that are open-ended. In the questionnaire, the scale "Strongly Disagree – Disagree – Neutral – Agree – Strongly Agree" and the numeric scale from 1 to 5 were interoperable, 1 being "Strongly Disagree" and 5 being "Strongly Agree" correspondingly. School student's answers can be observed in Appendix IV.

The average of all the Likert-scale questions answers is 4.69 (STD = 0.31), which, together with a low standard deviation, concludes that the module received positive feedback from the students. 18 students had an average response score over 4.00, and an average response score under 4.00 was observed with only 2 students, who had 3.94 (STD = 1.06) and 3.88 (STD = 0.89) average response scores respectively. Three questions were the main reasons for this to happen: "Working with the module content helped me learn", "I would like to get back to module content again" and "The module increased my interest in the IT field". It is interesting, however, that in the answers to the open-ended questions, both of them did not express any negative feedback, and vice-versa; one praised the atmosphere during the lesson (however, also stated that the module was difficult), and another mentioned homework assignments being interesting. The assumption can be made that these students were looking to try programming, and figured out that it's not something they would continue with, despite the module being still engaging when looking at the other answer.

First, the "Learning" section is analyzed. Overall, the average response score for this section is 4.65 (STD = 0.21). Students related the most to the "The module helped teach me a new concepts", with answer score averaging at 4.95 (STD = 0.22), while the weakest result was shown by "The graphics and animations from the module content helped me learn" at 4.30 (STD = 0.80). This is clear feedback that while already having a certain number of pop-culture references and analogies, more images, videos and overall visual content is something that students are looking for.

Second, the "Quality" section is analyzed. Overall, the average response score for this section is 4.65 (STD = 0.17). Students' feedback here was the most positive for the "The

additional explanations in the module content were useful" question, having its average response score of 4.85 (STD = 0.37), while the question "The instructions in the module were easy to follow" earned the least positive score in this section at 4.45 (STD = 0.69). This section contained questions about the learning setup (Notion, Replit, Discord) being "easy to use" and "well organized". Both questions resulted in 4.55 (STD = 0.69) for easiness of use and 4.75 (STD = 0.44) for being well organized.

Third, the "Engagement" section is analyzed. The section is averaging at 4.63 (STD = 0.24) across the three Likert-scale questions, with questions "I liked the overall theme of the module" and "I found the module motivating" standing at 4.80 (STD = 0.52) and 4.80 (STD = 0.62) respectively. The question "I would like to get back to module content again", however, earned the average response score of 4.30 (STD = 0.98), which, together with a high standard deviation, shows that this question divided those who liked the module and think of continuing their career in IT and those who despite liking the module, are not interesting in studying IT further.

The final two questions of this section were open-ended. The topics mentioned by the students in response to the "What, if anything, did you LIKE about the module?" were:

- Overall, the way the teacher conducted the lessons
- Teaching style and techniques (humor, analogies, atmosphere during lessons)
- Visit to the Delta Center of the University of Tartu
- Homework assignments
- Groupwork during the classes
- Final project in teams
- Guest lessons

The topics mentioned by the students in response to the "What, if anything, did you NOT LIKE about the module?" were:

- From "less time for easy topics" to "too complicated tasks"
- The short duration of the module
- More attention to explaining indent rules to student

One student mentioned that they would like to continue the module in the 12th grade, which would include more advanced topics. This was supported by another student who voiced their desire to have more content available.

Finally, the "Teaching" section is analyzed. The section is averaging at 4.83 (STD = 0.26), and is overall the most highly assessed section. The final question, "All in all, course was valuable for me" has gotten 4.95 (STD = 0.22) average response score, and the teaching quality questions "Teacher was providing helpful and sufficient feedback" and "Teaching was varied (different kinds of methods and tasks were employed)" earned a score of 4.90 (STD = 0.31) and 4.80 (STD = 0.41) respectively. The question that was aimed to discover whether the module influences students' career choices, "The module increased my interest in the IT-field" scored at 4.65 (STD = 0.81), having the least positive score in the section. This, similarly to the "I would like to get back to module content again" question from the last section, indicates that some student indeed increased their motivation to continue study IT, while for some the module was a pleasant experience of diving into IT and programming, not necessarily influencing their career choices.

5 Future Developments

Since generative AI tools have entered our daily and professional lives, the course needs a shift toward product-oriented thinking. In Table 4, one can see a plan of how the author sees the future of the IT module growing into the IT specialization ("*IT suund*" in Estonian), although still lightweight and adapted to a non-STEM language immersion school.

Table 4. The IT specialization dream

Course	Title	Key competences
1 (35 hours)	Business analysis	Validating the idea (competitor analysis, user interviews); Understanding the customer (creating user/buyer persona); Understanding the business model (Business model Canvas, pricing);
2 (35 hours)	Web-design	Designing a landing page (Figma); Making a landing page (HTML/CSS);
3 (35 hours)	Programming basics 1	Python programming essentials;
4 (35 hours)	Programming basics 2	Python programming essentials;
5 (35 hours)	IT Project	Developing a Discord-bot; Optionally, taking part in the Estonian student companies contest; Optionally, writing school's research work based on the project

The whole direction will need 5 courses, 175 academic hours in total, and will involve intense product-first thinking. The "IT Project" course can be integrated with the Estonian student companies contest, organized by Junior Achievement SA (https://ja.ee/). Students can then write the compulsory school research work ("*uurimistöö*" in Estonian) based on their final IT project, which is encouraged by the state curricula.

This way, this study direction will be nicely cross-integrated with other fields even further, strengthening cross-disciplinary knowledge exchange and improving students' overall skill-set and experience. The final IT project will still be a pretty backend-oriented chatbot-like solution since getting students ready for developing web applications is the next level of complexity and might require even more hours. However, teams will be tackled with the task of creating a static landing web page for their digital product. By keeping the direction simple enough, school students are taught all the tools to deliver tangible outcomes after each course and by the end of the study direction overall.

Conclusions

The aim of this thesis was to describe the motivation behind, the emergence, and the subsequent evolution of the IT module, a set of three courses designed by the author for Tartu Annelinna Gümnaasium. The motivation was the recognized absence of such a module in the school. This gap highlighted a need for a structured educational offering that could introduce school students to fundamental programming skills, provide insight into the IT industry, and offer career guidance. The gap was addressed by creating and implementing a module that would not only meet the educational needs of the students but would also align with the broader objectives of preparing them for future academic and professional pursuits in the field of computer science. The module's design is aligned with the Estonian state IT curriculum for upper-secondary schools, making sure to offer adapted, yet structurally similar courses to the ones recommended by the state. It takes into account Tartu Annelinna Gümnaasium context as a school that invests a large amount of academic hours into Estonian language immersion and has no STEM focus yet wants to keep a selection of the elective modules broad. Finally, the module is aligned with the University of Tartu, introducing Python programming language to prepare for the possible entrance exam and the first study year, and visiting the Delta Center of the University of Tartu during the "Delta day".

The IT module has been running as an elective module since 2020 and has experienced 4 iterations of continuous improvement. Action Research (AR) methodology is used to analyze the changes and what led to them. The module has thus evolved to be a well-established place in the school's curricula, a practice-proven and well-structured sequence of topics, the best-suiting digital learning environment setup (Discord, Replit, Notion), a set of effective teaching techniques (pop-culture references, analogies, real-life-inspired homework and incremental learning) and events (guest lessons, "Chasing Unicorns" movie session, the University of Tartu Delta center visit and Pipedrive office visit with pizza party for the high-achievers). The future developments include a plan for the module to transform into an intensely product-oriented set of 5 courses, adding business analysis and web design.

To validate the module's current state, the last iteration module graduates have been surveyed via an anonymous questionnaire aimed to assess the learning, quality and engagement aspects via Learning Object Evaluation Scale for Students (LOES-S) and teaching style and overall module value perception via questions from the University of Tartu Study Information System (SIS). The questionnaire results showed that students find the module engaging, useful and valuable. Certain answers indicated that there are two groups of students, those who see their future in the IT industry and those who don't, and both appeared to be content with the module experience regardless of their eventual career choice.

The module has multiple implications, highlighting a case of how Estonia's talent attraction programs via tuition-waiver university scholarships produce not only direct returns in the form of bringing a highly skilled IT specialist but also leading to indirect returns of the same IT specialist contributing back to the Estonian educational system. As a result, as of 2024 there are multiple students studying in the University of Tartu bachelor's programmes, who were IT module graduates.

In the feedback, the students praised the teaching style in particular, which is, while being positive feedback by itself, the author sees as a limitation, as human resources are hard to scale. Thus, the hope is for this thesis to be a detailed guide on how to scale the module's success. The module's materials are shared via a public link for fellow teachers and researchers.

Acknowledgments

I would like to express my sincere appreciation to my supervisors, Prof. Marina Lepp and Prof. Emanuele Bardone. Their guidance and expertise, along with their patience and mental support, were crucial in making this thesis possible. Thank you for believing in me and for pushing me forward.

I would also like to thank Prof. Marlon Dumas, the former Software Engineering program manager, who encouraged me to pursue studying abroad and provided support during challenging times in my life. I am grateful to Prof. Luciano García-Bañuelos, with whom I worked as a teaching assistant for several years. He has been a true inspiration of what a lecturer, mentor, and colleague should be: intelligent, empathetic, fun, and above all, a person with a big heart.

Special thanks go to Hiie Asser, the principal of Tartu Annelinna Gümnaasium, and Julia Trubatšova, the head of studies. Their openness to experimentation and innovation made the IT module possible. Their help and support during my teaching allowed me to focus on essential aspects and continuously improve the module.

My studies in Estonia were supported by the Estonian Foreign Ministry's Development Cooperation and Humanitarian Aid funds. I am deeply thankful to the Estonian state, its people, and the University of Tartu for the warm welcome I received. This thesis and the module it describes are just one of the many ways I express my gratitude and hope to contribute back to Estonian society.

On a personal note, I would like to thank my wife, Mariia, who not only supported me throughout my thesis work but also introduced me to Tartu Annelinna Gümnaasium when I first had the idea of a programming course. I am also grateful to my parents, Liudmyla and Oleksandr, who instilled in me a deep respect for academia and science, and who have always believed in my academic aspirations.

Finally, I would like to extend my thanks to all the school students who chose to spend 105 academic hours with me in their 11th grade, and a special shoutout to those who have successfully continued to study IT, whether at the University of Tartu or elsewhere. Keep up the great work!

References

Ceci, L. (2024, March 26). Most popular communication apps used by children in the United States in 2023. *Statista*. https://www.statista.com/statistics/1339198/leading-communication-apps-children-us/

Cohen, B. (2006). Sustainable valley entrepreneurial ecosystems. *Business strategy and the Environment*, *15*(1), 1-14.

DesPurpleLightning. (2020). Marriage Logic Map. *Reddit.* https://www.reddit.com/r/ProgrammerHumor/comments/jbw0hz/marriage_logic_map/

EIS. (2024, July 25). Testide tulemuste statistika. *EIS - Eksamide infosüsteem*. https://eis.ekk.edu.ee/eis/eksamistatistika

Fonseca Cacho, J. (2020). Using Discord to improve student communication, engagement, and performance. *Digital Scholarship @ University of Nevada, Les Vegas*. https://digitalscholarship.unlv.edu/cgi/viewcontent.cgi?article=1095&context=btp_expo

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, *42*(1), 38-43.

Honcharova, M. (2019). *Analyzing causes of differences in English language proficiency and academic performance in Estonian- and Russian-medium schools in Estonia* (Master thesis). University of Tartu. DSpace. https://dspace.ut.ee/items/23036d68-bd88-4727-8545-338832026ace

Izu, C., & Mirolo, C. (2024). Asking Students to Refactor their Code: A Simple and Valuable Exercise. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1* (pp. 73-79).

Januzi, S. (2020, November 4). Almost a Quarter of IT Employees in Estonia Are Foreigners. *Shengen News.* https://schengen.news/almost-a-quarter-of-it-employees-in-estonia-are-foreigners/

Kalda, K., Sell, R., & Soe, R. M. (2021). Use case of Autonomous Vehicle shuttle and passenger acceptance analysis. *Proceedings of the Estonian Academy of Sciences*, *70*(4), 429-435.

Kallas, K., Kaldur, K., Kivistik, K., Plaan, K., Pohla, T., Ortega, L., Mürk, I., Väljaots, K. (2014, April 9). Newly-arrived immigrants in Estonia: Policy Options and Recommendations for a Comprehensive and Sustainable Support System. *European Website on Integration*. https://migrant-integration.ec.europa.eu/sites/default/files/2014-04/docl_40861_917720677.pdf

Kay, R. H., & Knaack, L. (2009). Assessing learning, quality and engagement in learning objects: the Learning Object Evaluation Scale for Students (LOES-S). *Educational Technology Research and Development*, *57*, 147-168.

Kemmis, S., McTaggart, R., & Nixon, R. (2014). *The Action Research Planner: Doing Critical Participatory Action Research.* Springer.

Kori, K., Pedaste, M., Tõnisson, E., Palts, T., Altin, H., Rantsus, R., ... & Rüütmann, T. (2015, March). First-year dropout in ICT studies. In *2015 IEEE Global Engineering Education Conference (EDUCON)* (pp. 437-445). IEEE..

Laar, M. (2008). Leading a successful transition: the Estonian miracle. *European View*, *7*(1), 67-74.

Lees, M. (2016). Estonian education system 1990-2016: Reforms and their impact. *4Liberty.eu*. https://4liberty.eu/wp-content/uploads/2016/08/Estonian-Education-System_1990-2016.pdf

Leping, V., Lepp, M., Niitsoo, M., Tõnisson, E., Vene, V., & Villems, A. (2009, June). Python prevails. In *Proceedings of the International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing* (pp. 1-5).

Lepp, M., Luik, P., Palts, T., Papli, K., Suviste, R., Säde, M., & Tõnisson, E. (2017, June). MOOC in programming: A success story. In *Proceedings of the International Conference on e-Learning* (pp. 138-147).

Lin, A., & Cheung, T. (2014). Designing an engaging English language arts curriculum for English as a foreign language (EFL) students: Capitalizing on popular cultural resources. In *Popular culture, pedagogy and teacher education* (pp. 138-150). Routledge.

Lumi, L. J. (2024, July 19). Some Estonian university programs receive only 1 applicant for next academic year. *Eesti Rahvusringhääling*. https://news.err.ee/1609401283/some-estonian-university-programs-receive-only-1-applicant-for-next-academic-year

Mazina-Šinkar, J. (2022). *Academic Entrepreneurship: Facilitating Spin-Off Ventures In Estonian Universities* (Master's thesis). TalTech. https://digikogu.taltech.ee/en/Download/c0a85d8e-4161-4ce0-b4a7-64cb96da412b/Akadeemilineettevtlushargettevteteloomiset.pdf

Mehisto, P., & Asser, H. (2007). Stakeholder perspectives: CLIL programme management in Estonia. *International Journal of Bilingual Education and Bilingualism*, *10*(5), 683-701.

OECD. (2019). PISA 2018 Results (Volume I): What Students Know and Can Do. *OECD Publishing, Paris*.

OECD. (2023). PISA 2022 Results (Volume I): The State of Learning and Equity in Education. *OECD Publishing, Paris*.

Olesk, K. (2022, September 1). Topelt ei kärise! Ukraina noored jätkavad kooliteed nii Eestis kui ka kodumaal. *Postimees*. https://tartu.postimees.ee/7595883/topelt-ei-karise-ukraina-noored-jatkavad-kooliteed-nii-eestis-kui-ka-kodumaal

Ortega, L. (2014). *Highly-skilled migration: Estonia's attraction policy and its congruence with the determinants of 'talent mobility* (Master thesis). University of Tartu. DSpace. https://dspace.ut.ee/items/027136be-91ee-4c33-9ad1-cc969f878f58

Paulus, S. (2023, May). Estonia's HealthTech startup Better Medicine streamlines cancer diagnosis with AI to the rescue. *Invest in Estonia.* https://investinestonia.com/estonias-healthtech-startup-better-medicine-streamlines-cancer-diagnosis-with-ai-to-the-rescue/

Piltpull. (2021, August 16). Mäkk, hess voi miskit muud? *TikTok*. https://www.tiktok.com/@piltpull/video/6997017927312542981

Polidano, C., & Zakirova, R. (2011). *Outcomes from Combining Work and Tertiary Study. A National Vocational Education and Training Research and Evaluation Program Report*. National Centre for Vocational Education Research Ltd. PO Box 8288, Stational Arcade, Adelaide, SA 5000, Australia.

Salum, K., Luik, P., Lepp, M., & Laanpere, M. (2021). Teaching informatics in upper secondary school-preparing students for the future. In *ICERI2021 Proceedings* (pp. 3544-3553). IATED.

Saluveer, S.-K., & Truu, M. (2020, July). Startup Estonia White Paper 2021-2027. *Startup Estonia*. https://startupestonia.ee//wp-content/uploads/2023/11/SE_Whitepaper_Web-1-1.pdf

Sommer, S. (2020. February 11). Bolt kicks off self-driving technology research in partnership with the University of Tartu. *University of Tartu*. https://ut.ee/en/content/bolt-kicks-self-driving-technology-research-partnership-university-tartu

Stajano, F. (2000, January). Python in education: Raising a generation of native speakers. In *Proceedings of the 8th International Python Conference* (pp. 24-27).

Tartu Annelinna Gümnaasium. (2022, February 2). Tartu Annelinna Gümnaasiumi gümnaasiumiosa õppekava. *Tartu Annelinna Gümnaasium*. https://tag.ee/sites/default/files/gumnaasiumi_ok_23_02_2022.pdf

Teja, B. (2024, May 8). The Rise of Notion: From Scrappy Startup to a $10 Billion Valuation. *Feather*. https://feather.so/blog/notion-valuation

Tire, G. (2021). Estonia: A positive PISA experience. *Improving a Country's Education: PISA 2018 Results in 10 Countries*, 101-120.

Vabariigi Valitsus. (2011). Gümnaasiumi riiklik õppekava. *Riigi Teataja*. https://www.riigiteataja.ee/akt/108032023006

Velt, H., Torkkeli, L., & Saarenketo, S. (2018). The entrepreneurial ecosystem and born globals: The Estonian context. *Journal of Enterprising Communities: People and Places in the Global Economy*, *12*(2), 117-138.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Xu, W., Wu, Y., & Ouyang, F. (2023). Multimodal learning analytics of collaborative patterns during pair programming in higher education. *International Journal of Educational Technology in Higher Education*, *20*(1), 8.

Yang, Y., Hooshyar, D., Pedaste, M., Wang, M., Huang, Y. M., & Lim, H. (2020). Prediction of students' procrastination behaviour through their submission behavioural pattern in online learning. *Journal of Ambient Intelligence and Humanized Computing*, 1-18.

Appendix

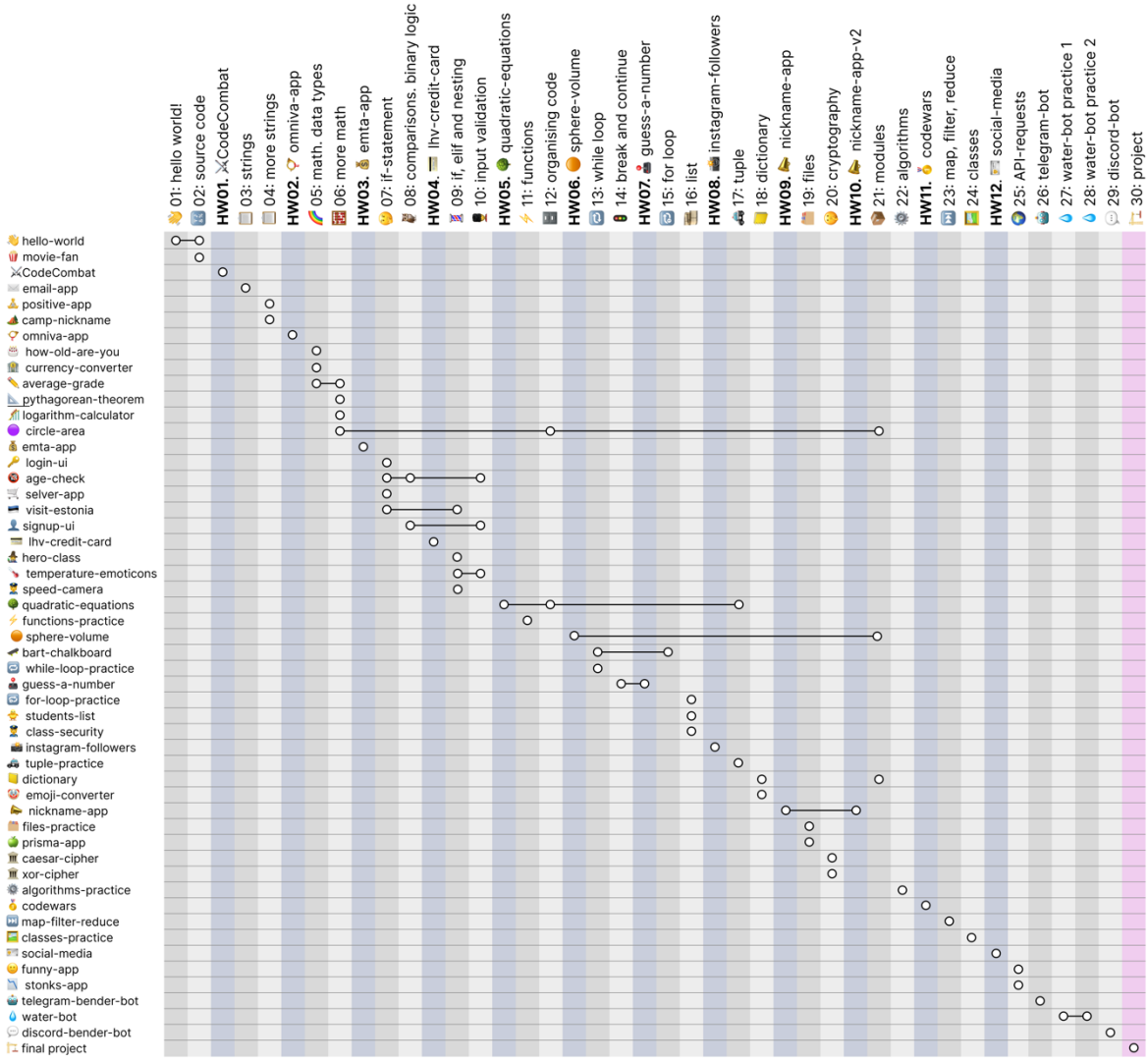
I. Lessons, tasks and homework assignments

The IT module has 30 topics, 12 homework assignments and the final project.

Lessons	IT module courses		
	Programming basics 1	**Programming basics 2**	**IT project**
1-2	01: hello world!	15: for loop	30. Project intro
3-4	02: source code	16: list	Brainstorm
Homework	HW01. codecombat.com	HW08. instagram-followers	-
5-6	03: strings	17: tuple	Ideas pitch
7-8	04: more strings	18: dictionary	Ideas validation
Homework	HW02. omniva-app	HW09. nickname-app	-
9-10	05: math. data types	19: files	Requirements
11-12	06: more math	20: cryptography	Requirements
Homework	HW03. emta-app	HW10. nickname-app-v2	-
13-14	07: if-statement	21: modules	Development
15-16	08: binary logic	22: algorithms	Development
Homework	HW04. lhv-credit-card	HW11. codewars.com	-
17-18	09: if, elif and nesting	23: map, filter, reduce	Development
19-20	10: input validation	24: classes	Development
Homework	HW05. quadratic-equations	HW12. social-media	-
21-22	11: functions	25: API-requests	Development
23-24	12: organising code	26: telegram-bot	Development
Homework	HW06. sphere-volume	-	-
25-26	13: while loop	27: water-bot practice 1	Development
27-28	14: break and continue	28: water-bot practice 2	Development
Homework	HW07. guess-a-number	-	-
29-30	"Üksarvik" movie party	29: discord-bot	Final demo
31-32	consultation	consultation	consultation
33-34	consultation	consultation	consultation

II. Topics and tasks

The IT module has 53 tasks (40 classroom tasks, 12 homework assignments and 1 final IT project), connected across 30 topics via incremental learning. Sometimes the task can be introduced in one topic and later be requested to be refactored in the following topic.



III. Feedback questionnaire questions

Original questions	Modified question	Scale	Source
Learning			
Working with the learning object helped me learn	Working with the module content helped me learn	Likert (1-5)	LOES-S
The feedback from the learning object helped me learn	The feedback (error messages from Replit) helped me learn	Likert (1-5)	LOES-S
The graphics and animations from the learning object helped me learn	The graphics and animations from the module content helped me learn	Likert (1-5)	LOES-S
The learning object helped teach me a new concept	The module helped teach me a new concept	Likert (1-5)	LOES-S
Overall, the learning object helped me learn	Overall, the module helped me learn	Likert (1-5)	LOES-S
Quality			
The help features in the learning object were useful	The additional explanations in the module content were useful	Likert (1-5)	LOES-S
The instructions in the learning object were easy to follow	The instructions in the module content were easy to follow	Likert (1-5)	LOES-S
The learning object was easy to use	The learning setup (Notion, Replit, Discord) was easy to use	Likert (1-5)	LOES-S
The learning object was well organized	The learning setup (Notion, Replit, Discord) was well organized	Likert (1-5)	LOES-S
Engagement			
I liked the overall theme of the learning object	I liked the overall theme of the module	Likert (1-5)	LOES-S
I found the learning module motivating	I found the module motivating	Likert (1-5)	LOES-S
I would like to use the learning object again	I would like to revisit module content some time	Likert (1-5)	LOES-S
What, if anything, did you LIKE about the learning object	What, if anything, did you LIKE about the module	Open question	LOES-S
What, if anything, did you NOT LIKE about the learning object	What, if anything, did you NOT LIKE about the module	Open question	LOES-S
Teaching			
Teaching was varied (different kinds of methods and tasks were employed)	I liked the overall theme of the module	Likert (1-5)	SIS
The course was intellectually challenging	I found the module motivating	Likert (1-5)	SIS
The course increased my interest in the field	The module increased my interest in the IT field	Likert (1-5)	SIS
All in all, the course was valuable for me	All in all, the module was valuable for me	Likert (1-5)	SIS

IV. Feedback questionnaire results

The answers to the open-ended questions, with the average score for the Likert scale questions and standard deviation provided for every responder:

N	AVG	STD	What, if anything, did you LIKE about the module?	What, if anything, did you NOT LIKE about the module?
1	5.00	0.00	*УЧИТЕЛЬ!!!! Подача материала и тема курса*	*Домашние задания слишком быстро выполнялись и хотелось больше задачек что бы выполнять их дома*
2	3.94	1.06	*офигенный вайб уроков*	*много сложностей*
3	4.69	0.48	*Мне понравился сам процесс написания кода, когда то, что ты создаёшь своими руками, оживает.*	-
4	4.94	0.25	-	-
5	4.88	0.34	*Легкое объяснение всего материала, разнообразие задач*	*Слишком мало времени длился🤣*
6	4.31	0.87	-	-
7	4.75	0.58	*абсолютно ВСЕ!*	-
8	4.88	0.34	*Юмор преподавателя, умение замотивировать, помощь в любое время суток, не большая нагрузка, понятные аналогии для учеников, подробное описание задания*	*Одного модуля мало*
9	4.81	0.40	-	-
10	3.88	0.89	*Домашние задание были, очень интересные*	*Не помню)*
11	4.69	0.70	*Visiting the delta showed where I could study in the future.*	*I need mooore content, maybe a continuation of the module in the next 12 class, or a separate module on a different topic related with programming.*
12	4.75	0.45	*Уроки с гостями, стиль ведения уроков*	*Всё понравилось)*
13	4.94	0.25	*Подход учителя к обучению*	*Ничего*
14	5.00	0.00	*Understanding what IT is and gaining knowledge in the field required not only theoretical knowledge but also practical experience. Throughout the learning process, I discovered what skills and knowledge are necessary for a career in IT and where I could gain this experience.* *A key part of this process was my teacher. He was not only a very pleasant person but also an excellent educator. His kindness, positive attitude, and motivating atmosphere greatly contributed to effective learning. Thanks to his support and teaching methods, the learning experience was both simple and highly effective.* *Most importantly, I was able to understand that I truly want to pursue a career in IT. This experience showed me that the field of information technology is where I see my future.*	*I liked everything*
15	4.69	0.48	*Подача материала*	
16	4.88	0.34	*Интересные задания*	*Иногда необходимость ждать урок или 2 до модуля*
17	4.69	0.48	*Темы, затронутые в модуле; преподаватель; обратная связь; совместная работа с преподавателем и одногруппниками*	*Нет того, что мне не понравилось.* *Может только стоит по ситуации уделять легким темам меньше времени, чтобы на сложные темы было больше времени для объяснений и вопросов (смотря как быстро ученики усваивают инфу).*
18	4.88	0.34	*Организация уроков, способ подачи материала при помощи наглядности и практики*	-
19	4.56	0.63	*Обучение чему-то новому*	*Не объяснили, как правильно оформлять код(пробелы, отступы)*
20	4.69	0.60	*I had a lot of interesting things in this module, but I was most pleased with the final task where we wrote a bot in groups*	*I am happy with everything*

Here are the answers to the Likert-scale questions, with the calculated average and standard deviation for every responder, question and section:

Questions	N																					
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVG	STD
Learning																					**4.65**	**0.21**
Working with the module content helped me learn	5	3	5	5	5	4	5	5	5	3	5	5	5	5	4	5	5	5	5	5	4.70	0.66
The feedback (error messages) from Replit helped me learn	5	4	5	5	4	5	4	4	4	4	5	5	5	5	4	5	4	5	5	5	4.60	0.50
The graphics and animations from the module content helped me learn	5	2	4	5	4	5	5	4	4	5	5	5	4	5	5	4	4	4	4	3	4.30	0.80
The module helped teach me a new concepts	5	5	5	5	5	5	5	5	5	4	5	5	5	5	5	5	5	5	5	5	4.95	0.22
Overall, the module helped me learn	5	4	4	5	5	5	5	5	5	3	5	5	5	5	4	5	5	5	4	5	4.70	0.57
Quality																					**4.65**	**0.17**
The additional explanations in the module content were useful	5	5	5	5	5	5	5	5	5	4	5	5	5	5	4	5	5	5	4	5	4.85	0.37
The instructions in the module were easy to follow	5	3	5	5	5	4	4	5	5	5	5	4	5	5	5	4	4	4	3	4	4.45	0.69
The learning setup (Notion, Replit, Discord) was easy to use	5	3	4	5	5	4	5	5	4	5	3	5	5	5	4	5	5	5	5	4	4.55	0.69
The learning setup (Notion, Replit, Discord) was well organized	5	4	4	5	5	5	5	5	5	5	4	4	5	5	5	5	5	5	5	4	4.75	0.44
Engagement																					**4.63**	**0.24**
I liked the overall theme of the module	5	4	5	5	5	3	5	5	5	4	5	5	5	5	5	5	5	5	5	5	4.80	0.52
I found the module motivating	5	5	5	5	5	3	5	5	5	3	5	5	5	5	5	5	5	5	5	5	4.80	0.62
I would like to get back to module content again	5	2	5	4	5	3	3	5	5	3	3	4	5	5	5	5	5	5	4	5	4.30	0.98
Teaching																					**4.83**	**0.26**
Teaching was varied (different kinds of methods and tasks were employed)	5	5	4	5	5	5	5	5	5	4	5	5	5	5	5	5	4	5	4	5	4.80	0.41
Teacher was providing helpful and sufficient feedback	5	5	5	5	5	5	5	5	5	4	5	5	5	5	5	5	4	5	5	5	4.90	0.31
The module increased my interest in the IT-field	5	4	5	5	5	3	5	5	5	2	5	4	5	5	5	5	5	5	5	5	4.65	0.81
All in all, course was valuable for me	5	5	5	5	5	5	5	5	5	4	5	5	5	5	5	5	5	5	5	5	4.70	0.66
	5.00	3.94	4.69	4.94	4.88	4.31	4.75	4.88	4.81	3.88	4.69	4.75	4.94	5.00	4.69	4.88	4.69	4.88	4.56	4.69	**4.69**	
	0.00	1.06	0.48	0.25	0.34	0.87	0.58	0.34	0.40	0.89	0.70	0.45	0.25	0.00	0.48	0.34	0.48	0.34	0.63	0.60	**0.31**	

Distribution of average scores per each section:

- Learning section: **4.65** (STD = 0.21)
- Quality section: **4.65** (STD = 0.17)
- Engagement section: **4.63** (STD = 0.24)
- Teaching section: **4.83** (STD = 0.26)

The module's total average score is **4.69** (STD = 0.31) among all the students' scores.

V. License

Non-exclusive licence to reproduce the thesis and make the thesis public

I, Mykhailo Dorokhov,

1. grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright, my thesis **"Python programming module for non-specialized schools as an introduction to IT"** supervised by Marina Lepp and Emanuele Bardone.

2. I grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in points 1 and 2.

4. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Mykhailo Dorokhov

13.08.2024