CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Impact of Generative AI on Learning Programming

Master's thesis in Computer science and engineering

KENNY BANG
MICHAEL DANG

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Impact of Generative AI on Learning Programming

KENNY BANG
MICHAEL DANG

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Impact of Generative AI on Learning Programming

KENNY BANG MICHAEL DANG

Supervisor: Philipp Leitner, Computer Science and Engineering
Examiner: Farnaz Fotrousi, Computer Science and Engineering

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Impact of Generative AI on Learning Programming

KENNY BANG
MICHAEL DANG
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Generative AI has become a powerful tool, particularly in code generation for non-programmers. Despite its potential, non-experienced programmers may face challenges and limited knowledge acquisition.
This study analyzes whether generative AI, specifically ChatGPT, can assist individuals with limited programming knowledge in solving programming tasks in Java and if they learn from the experience. Additionally, it examines the strategies participants use when leveraging ChatGPT 3.5.
A controlled experiment was conducted to observe the effects of ChatGPT on problem-solving and learning. Participants completed three programming tasks: a baseline task, a task using ChatGPT, and a final task without ChatGPT to assess knowledge retention. This revealed varying degrees of success in task completion with ChatGPT. While participants retained conceptual knowledge, practical application without ChatGPT remained challenging.
Thematic analysis of participant prompts and exit interviews revealed three main strategies: learning-focused, task-focused leveraging ChatGPT, and conversational interaction with ChatGPT. The learning-focused strategy involves using the AI to gain knowledge but may be ineffective if rushed, hindering comprehension. In contrast, the task-focused strategy prioritizes task completion but may lead to over-reliance on AI, limiting long-term learning. The conversational strategy is distinct because it does not serve a specific purpose like the other two. Instead, it involves a back-and-forth interaction with the AI, where participants build upon ChatGPT's previous responses. Overall, participants had a positive attitude towards generative AI, recognizing both its benefits and limitations. The study highlights the importance of prior knowledge and effective prompt engineering to formulate informed prompts and obtain quality responses, thereby improving the overall experience with AI.

Keywords: generative AI, problem-solving, learning, engineering, prompt engineering, knowledge retention, Large Language Model (LLM), ChatGPT.

Acknowledgements

We would like to extend our deepest gratitude to our supervisor, Philipp Leitner, for their invaluable guidance and support throughout the course of our research. We would also like to thank everyone who participated and contributed to both our pilot and main studies.

Kenny Bang & Michael Dang, Gothenburg, June 2024

Contents

List of Figures

List of Tables

1

Introduction

Generative AI has emerged as a powerful tool and is changing the landscape of AI applications. Exemplified by models such as GPT (Generative Pre-trained Transformer), generative AI is distinguished by its creativity. As mentioned by Martineau [1], these systems can produce diverse outputs, such as texts and images, to create content not explicitly present in their original training data. One specific application is code generation, which enables non-programmers to write code and experienced programmers to streamline software development. As argued by Peng et al. [2], AI can help in finishing programming tasks about twice as fast as without it.

While it is important to understand how powerful generative AI can be, it is also crucial to understand the limitations. The AI tool is only as good as the data it has been trained on, and as highlighted by Qadir, the AI can give answers that are biased and fabricated [3]. Furthermore, as expressed by Qadir, "effectiveness of any resource or tool is limited by an individual's existing knowledge and background". This implies that users with limited knowledge are likely to face challenges in solving complex tasks, with the probability of acquiring new knowledge being even lower. That is because the individual would still need to provide the AI with relevant contextual questions to guide it toward generating a helpful solution. Therefore, it is essential for programmers to be able to think critically even when using generative AI such as ChatGPT. Another important aspect of generative AI is how users interact with and leverage the AI. Various factors play into the effectiveness of these tools. How a user frames their prompts can impact the relevancy and quality of the generated content. Beyond that, users must be able to interpret and evaluate the response generated, discerning their accuracy and deciding whether to rely on the outputs or seek additional information. This is especially important when integrating the generated content into workflows or projects. Often requiring refinement or the incorporation of other information sources to achieve the desired outcomes, which puts a demand on the user, even when employing advanced generative AI models.

The goal of this study is to analyze and comprehend the extent to which generative AI can help someone with limited knowledge to solve a complex coding task, and whether they learn anything from it. Furthermore, the study aims to provide insights into the strategies and approaches programmers use when leveraging these tools to enhance their problem-solving capabilities.

1.1 Problem Statement

Skilled developers excel in solving coding problems by employing strategical approaches. Using generative AI, code generation and code development is made accessible to non-developers, as mentioned by IBM Education [4]. From this, AI has the possibility to give non-experienced programmers the ability to start coding as a developer. Guardrails [5] bring up that this can be beneficial for learning programming, since AI can lower the initial learning curve for programming. However, overreliance on AI without understanding underlying principles, might result in issues solving basic coding tasks. Furthermore, it may also hinder the ability to troubleshoot when bugs and issues arise. This problem can also affect experienced programmers who are exploring new domains, in which you miss out on learning and understanding fundamental concepts. It might also discourage programmers from developing problem-solving skills and the ability to work through coding challenges independently.

As highlighted by Qadir [3], we observe that there is acknowledgment that generative AI has both positives and negatives. There is merit in conveying the potential impact, and making people question and use AI with more thoughtfulness. In addition, Michel-Villarreal et al. [6] sought to identify shortcomings of generative AI and try to create "mitigation strategies for addressing the identified challenges". Even then, there is no way of enforcing these guidelines. Generative AI is readily available on the internet and anyone can use it.

Instead of identifying weak points of AI and suggesting guidelines, the study will try to put "using generative AI" in practice and see if it actually impacts programming and focusing in on the aspect of learning. Furthermore, with the emergence of relatively new generative AI tools such as ChatGPT, there exist a notable research gap regarding the awareness of proper user strategies and approaches to utilize such tools within education and professional workspaces [7]. Therefore, this study also aims to explore the strategies and approaches that users, especially learners, employ when interacting with generative AI. Analyzing and understanding how programmers engage with and utilize AI tools in their learning process can be crucial when it comes to evaluating its influence on learning outcomes and effectiveness.

1.2 Purpose of the Study

The purpose of this study is to explore how generative AI affects programming. More specifically, it investigates whether a non-programmer can solve programming problems with AI and subsequently retain and use this knowledge independently, without the aid of AI. Additionally, it examines the strategies and approaches they use, along with their advantages and limitations.

By putting ChatGPT 3.5 into practice and observing if knowledge is gained from solving coding tasks with AI, we can provide insights on how to use it better and whether it is beneficial or detrimental to use for learning. Since generative AI is fairly new, there might be mixed thoughts on its adoption. Some might restrict its use, fearing its shortcomings, while other employ it without caution. We will

try to pinpoint if it is wise to work with generative AI when programming. The research should advise people who are learning programming, such as students, but also experienced and professional programmers, who are delving into new domains and languages, whether they will gain any knowledge from its use. Educators can also use our findings to understand the impact of AI on learning, helping them create clear guidelines for students on its use in studies.

1.3 Research Questions

The first research question is:

RQ 1: Can a person with limited programming experience successfully solve a coding task with the assistance of AI?

- **RQ 1.1:** If yes, what programming knowledge does the subject acquire during the process?
- **RQ 1.2:** If no, does the subject still acquire any programming knowledge during the attempt?

Hypothesis RQ 1:

- **Null Hypothesis (H0):** There is no difference in the ability of beginner programmers to solve coding tasks with or without the use of AI.
- **Alternative Hypothesis (H1):** Beginner programmers are more likely to solve coding tasks when using AI compared to when they do not use AI.

The null hypothesis assume that the use of AI has no effect on the ability of beginner programmers to solve coding tasks. On the other hand, the alternative hypothesis suggests that the use of AI improves beginner programmers ability to solve coding tasks.

Our initial expectation leans towards the null hypothesis (H0). This expectation is based on the premise that the quality of AI assistance depends significantly on the clarity and specificity of the questions posed. As articulated by Qadir [3], "The ones who can question well have a great advantage as getting answers is free and the quality of the answers depends on the quality of the questions." This emphasizes the importance of writing proper prompts to receive better answers. Consequently, beginners may struggle to benefit from AI assistance if they lack the ability to formulate effective prompts.

The second research question is:

RQ 2: What are the strategies and approaches employed by beginner programmers when utilizing generative AI assistance for programming tasks?

- **RQ 2.1:** What are the benefits and challenges associated with this usage?

1.4 Significance of the Study

The study aims to present insights into the impact of generative AI on programming and whether any knowledge gained can be utilized without relying on it. Additionally, it aims to provide further theoretical development in the domain of strategies and approaches for generative AI usage. The contributions provide educators, students and professionals with an awareness of the impact that AI has on learning,

allowing them to make informative decisions on whether to use it or not. The study also contributes to the vast and current research on generative AI and its effects on the world.

1.5 Disposition of the Report

Section two introduces the related work and recent findings in the field. Section three discusses the study design and the process of making it. Section four presents the results from the study. Section five discusses the results. Section six summarizes the findings with conclusions and propose future works.

2
Related Work

As a field of computer science, artificial intelligence has a long history of continuous innovation and exploration. Even though this technology has been evolving and advancing for many years, the release of ChatGPT represents a major milestone in the landscape of AI innovation. Despite its large-scale breakthrough, there remains an extensive amount of unknowns, especially when it comes to its application in education and learning. Additionally, there are still uncertainties and no clear guidelines regarding the strategies and approaches on how to use AI for programming. The following section explores existing literature surrounding the opportunities and challenges emerged by AI and ChatGPT in this domain.

2.1 The Impact on Education

The rise of new technologies such as ChatGPT has the potential to disrupt traditional practices, particularly in the context of education. There will be a need for people to adapt to new technologies and to consider the potential benefits and drawbacks of them, and whether if these tools make us more knowledgeable or superficial. This type of technology can pose a risk of job displacement, particularly for low-skilled workers. Therefore, it's important to establish and develop clear guidelines on how to use such tools so that the engineering education community can benefit from it while also mitigating the potential downsides [3].

The prevalent use of ChatGPT in education has raised concerns about its potential impact on students' critical thinking and problem-solving skills [8]. As previously mentioned, ChatGPT is a powerful tool that is able to generate code and provide information with flexibility in adapting to different contexts. An overreliance on it may hinder the development of critical and analytical thinking within students and programmers. Humble et al. [9] warn teachers that ChatGPT can potentially obstruct learning and that it is possible to utilize it for cheating and hide a lack of understanding. Since the content produced by ChatGPT is unique, it can pose a challenge for educators in terms of assessing the knowledge of students as the result of evidence that text generated by ChatGPT can bypass traditional plagiarism detectors, making it difficult to distinguish work that has been created by students themselves and generative AI. Humble also mentions that the uniqueness of ChatGPT's answers makes it difficult to determine the general quality of ChatGPT's responses.

Personalized learning is a topic that is frequent in the research of AI on education. Baidoo-anu [10] and Alasadi [11], mentions that AI can be used to personalize

the learning experience, taking into consideration the user's current strengths and weaknesses, as well as their needs and progress. This enables students to receive guidance and material that is adapted to their level of understanding, helping them engage better with the learning material. However, Baidoo-anu also mentions that generative AI models are based on the data they are trained on, and therefore do not have a proper understanding that they are helping students learn. Consequently, this can lead to a failure to consider a student's needs. Additionally, there is not enough empirical evidence whether this personalized tutoring is able to enhance the learning experience.

2.2 Effectiveness of ChatGPT in Learning Programming

In a related study by Rahman and Watanobe [12], the impact of ChatGPT on programming learning was explored and while the study acknowledged ChatGPT's potential in assisting students with complex tasks, it primarily focused on qualitative assessments through interviews with students and teachers, which did not directly address the effectiveness of learning programming with ChatGPT. The interview questions such as "Have you taken help/support from ChatGPT for solving programming problems?" and "Do ChatGPT's suggestions help you in solving programming problems?" mainly assess the students' subjective opinions and experiences rather than objective measures for learning outcomes and the quality of learning achieved through those interactions. This further suggests a gap in the literature regarding empirical evidences into the impact of generative AI on learning programming.

Jing et al. [13], conducted an experimental study about what factors influences the effectiveness of using ChatGPT to solve programming tasks. The study examines various aspects such as the cognitive abilities of learners and how some factors, most notably AI literacy and the base knowledge of the programming language used in the study, affect the effectiveness of ChatGPT usage. The recruited participants were students from a university that were instructed to solve a programming task. The study offered valuable insights into the factors influencing the effectiveness of using ChatGPT. Furthermore, the authors discussed the limitations and proposals for future research such as that "the study lacked a focus on the efficiency of learners in using ChatGPT" and "subsequent research is needed to expand beyond the Python language ... as well as consider variations in experimental task types".

In a study by Yilmaz and Karaoglan Yilmaz [14], the authors explored the use of ChatGPT in programming learning among undergraduate students. Over an eight-week period, the students were given weekly assignments in an Object-Oriented Programming course, where the students were able to utilize ChatGPT to complete the assignments. An analysis of student feedback showed both benefits and limitations of ChatGPT. The majority of the students expressed that the AI tool enhanced their cognitive abilities "By reducing the time spent on coding tasks, students are able to allocate more time towards engaging in algorithmic thinking processes for effective problem-solving" which was a result of ChatGPT giving them quick and

effective answers. However, other students stated that these fast and accurate responses could harm and hinder development of their thinking skills. Furthermore, the authors examined current existing literature, Tlili et al. [15] and Yilmaz and Karaoglan Yilmaz [16] which was consistent with the findings of their study that ChatGPT and AI-based personalized tools can enhance students cognitive abilities and learning motivation. Additionally, the authors end their study with "To determine the effect of generative AI tools such as ChatGPT on students' learning processes and outcomes, it would be useful to plan experimental studies, especially with experimental and control groups" which further suggest a research gap when it comes to the impact of generative AI on learning programming.
As a part of a study by Popovici [17], 181 students volunteered to answer a survey regarding generative AI. The survey showed that more than 40% of the participants had used generative AI for coursework or other school activities. However, the most interesting result from the survey is that about 55% of the students answered yes to a various extent when asked "has generative AI helped you gain a better understanding of curricula?" which shows the potential that generative AI can facilitate learning.
As observed in the studies mentioned above, investigations into ChatGPT's impact on learning collectively underscore both the promise and the need for further empirical research in understanding the role of generative AI in programming education.

2.3 Strategies and Approaches

As previously mentioned, ChatGPT is a relatively novel technology, consequently the established strategies and approaches regarding its optimal utilization are inadequate, particularly in the domain of programming. However, one developing concept is **prompt engineering**, which directly influences how effectively users can leverage ChatGPT for programming tasks. As explained by White et al. [18], "prompt engineering is an increasingly important skill set needed to converse effectively with large language models (LLMs), such as ChatGPT". Moreover, the authors present a catalog of prompt patterns, which goal "is to enhance prompt engineering by providing a framework for designing prompts that can be reused and/or adapted to other LLMs, in the same way that software patterns can be implemented in different programming languages and platforms". These prompt patterns were classified and put in different pattern categories for better documentation. For instance, the Error Identification category includes patterns like Fact Check List, which requires the LLM to generate a list of facts that the output depends on, and Reflection, which prompts the LLM to self-reflect on its outputs and identify any errors. This strategy provide a user with a way to systematically identify and resolve errors in their code, potentially leading to better learning outcomes and higher-quality code. Similarly, the Prompt Improvement category aims to enhance the quality of both input and output. Patterns like Question Refinement, which ensures that the LLM suggests improved alternatives to the user's question, and Alternative Approaches, which requires the LLM to suggest different methods to accomplish a given task, can help users refine their queries and explore multiple solutions to programming problems. The Interaction category focuses on the interaction between the user and the LLM. Patterns such as Flipped Interaction, which requires the LLM to ask

questions rather than generating outputs, and Game Play, which requires the LLM to generate answers in the form of a game, offer innovative ways to engage users. These interaction strategies can deepen the engagement the user has with the LLM and can facilitate learning.

Wei et al. [19], describe chain-of-thought prompting as "a series of intermediate natural language reasoning steps that lead to the final output". This approach mirrors the human cognitive process when it comes to problem-solving by prompting the model to generate intermediate reasoning steps before arriving at the final answer. Similarly, Zhuo et al. [20] present a strategy called "least-to-most prompting" which involves breaking down a complex problem into simpler subproblems and solving them sequentially. Additionally, Zhuo et al. explains, "Solving each subproblem is facilitated by the answers to previously solved subproblems". While both of these strategies involve breaking down the task into manageable steps, Zhuo et al. states that experimental results have shown that least-to-most prompting outperforms both standard prompting and chain-of-thought prompting in tasks that require more complex reasoning.

Despite the existence of these strategies for effectively using LLMs, there is limited focus on how beginner programmers use ChatGPT. Investigating this can reveal how novices interact with the tool and whether they naturally employ these established strategies or develop new ones. By examining the strategies used by participants and comparing them to the established ones, we can gain insights into which approaches are most effective. Analyzing the use of these strategies among participants can provide valuable information on effective practices and highlight the practical implications of these strategies. This can lead to improved knowledge of the strategies used, encouraging users to employ them to enhance their learning outcomes and programming proficiency. As a result, it can contribute to the development of best practices for using ChatGPT in educational settings.

3

Methods

This section outlines the methodology employed to address the research questions. Figure 3.1, inspired by Jing et al. [21], presents the major components of the study. It starts with an overview of the study design and participant selection process. Then, it elaborates on the development of coding tasks and the pilot study conducted to identify and address potential issues. Following this, the procedural framework adopted for the study is presented. Finally, the section shows the data collected, including the analysis procedures.

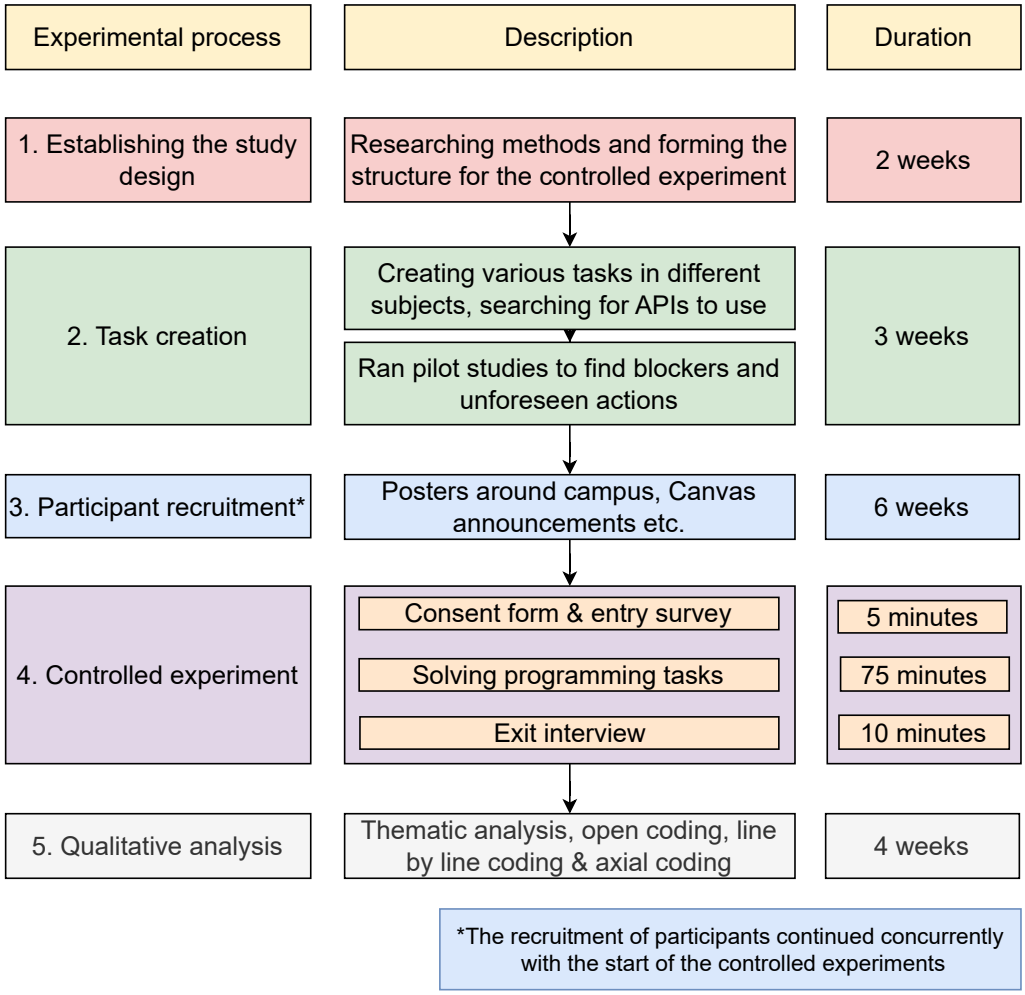


Figure 3.1: Study design process

3.1 Establishing the Study Design

In the initial stages, it was determined that a **controlled experiment** would be conducted to investigate the impact of generative AI on programming, addressing the research questions. A controlled experiment is a scientific study in which "an intervention is deliberately introduced to observe its effects" as mentioned by Fittkau [22]. This research design fit the purpose of the paper with the intervention being ChatGPT 3.5, and seeing if participants successfully solve coding tasks. ChatGPT 3.5 was chosen because it was the latest version with the most availability due to being free.

3.1.1 Participants

The main goal was to select participants that were bachelor's students currently studying in the field of Computer Science, Software Engineering, or a related program at Chalmers University of Technology and University (CTH) of Gothenburg (GU). This selection aimed to ensure a baseline level of programming while still allowing for potential learning effects of generative AI to be observed. However, due to the limited number of available participants, we decided to also include master's students to increase the sample size, despite their potentially more advanced programming skills. Additionally, it is worth noting that not all students have the same level of programming knowledge. Some may have programmed before starting their bachelor's degree, while others might program frequently in their free time. For some, their bachelor's program was the first time they were exposed to programming. While the aim was to include participants with similar programming knowledge, it is recognized that varying skill levels are inevitable. Given the constraint of participant availability, there was no room for selectivity in this regard. To address the variability in programming knowledge among participants, an entry survey was added to record their current programming knowledge and experience. This additional data point would enable accounting for any disparities in programming when analyzing the results. In Table 3.1 are the participants that took part in the study along with their prior experiences.
Another anticipated challenge was participant recruitment, particularly due to the lack of incentives. Consequently, the duration of the study required careful consideration. A shorter study might attract more participants due to reduced time commitment, but it could compromise the depth of participant engagement with the programming tasks. On the other hand, a longer study might deter potential participants. In a survey of controlled experiments conducted by Sjoeberg [23], it was found that most experiments did not exceed two hours in duration. While two hours represent only a fraction of the time typically spent on learning and programming, the feasibility of conducting a longer study had to be carefully considered in light of participant willingness.
The recruitment process for participants mainly consisted of setting up posters around the campuses and sending out announcements to all the relevant students through Canvas, the learning management system used by both Chalmers and GU. Additionally, our supervisor informed his students during one of their lectures about

Participant	Program	Before[1]	Beside[2]	Java	Web API	ChatGPT
1	N1SOF (GU)	-	-	+	-	+
2	N1SOF (GU)	+	+	+	+	+
3	N1SOF (GU)	-	-	+	-	+
4	N1SOF (GU)	-	-	+	-	+
5	TIDAL (CTH)	-	+	+	-	+
6	N1SOF (GU)	-	-	+	-	+
7	MPALG (CTH)	+	-	+	-	+

1. Programming before university
2. Programming beside university

Table 3.1: Participants' programs and whether they have programming experience before, beside, and during university, experience with Java and Web API programming, and whether they have used ChatGPT. Programs: N1SOF - Software Engineering and Management Bachelor's Programme, TIDAL - Computer Science and Engineering Bachelor's Programme, MPALG - Computer Science – Algorithms, Languages and Logic Master's Programme.

our study, which most of our participants ended up coming from. Despite these widespread efforts, the final sample size of seven, was deemed insufficient to primarily focus on a quantitative analysis. Consequently, **RQ 2** and the thematic analysis emerged, prioritizing the qualitative aspect of the study.

3.1.2 Within-subjects

In response to the challenge of participant recruitment, a within-subjects design was chosen. This experimental approach, as described by Blandford [24], involves each participant engaging with all the set conditions, solving tasks with and without ChatGPT. This is in contrast to between-subjects design, which requires different groups to be tested on each condition. This choice was motivated by the need to minimize participant burden and maximize efficiency. The between-subjects design demands more participants since each group requires sufficient participants on each side to discern meaningful differences as mentioned by Raluca [25]. A within-subjects design allowed for comparison of conditions with the same group, which would prove advantageous if the number of participants were lower.

3.1.3 Interview

Lastly, an interview at the end of the study was added for triangulation and enhancing validity. The interview would look into whether participants had learned anything and what their approaches were using ChatGPT. It was also an opportunity to question any ambiguity observed during the study, but also as a moment for the participant to freely speak about their experience to help them feel valued at the end of the study as according to Blandford [24].
The interview had a semi-formal structure with a set of questions, but allowed for other questions to be brought up as well. The set questions were:

- What are your general thoughts on the experiment?
- What difficulties did you stumble upon?
- What are your overall thoughts on the use of ChatGPT, in assisting with programming tasks?
- What did you learn?
- How helpful was ChatGPT?
- How did you use ChatGPT to help you with the task?

3.2 Task Creation

The study comprises three phases, and for each phase, there is one task to solve. The first phase serves to establish a baseline of a participant's knowledge without the assistance of ChatGPT. If we were to begin with a task using ChatGPT, it would be harder to conclude whether the solution derived from the participant's knowledge or from the capabilities of the AI. The second phase allows participants to use ChatGPT during the task. Finally, the third phase, conducted without ChatGPT, to see if there is any knowledge acquisition or changes in approach following the interaction with the AI. The latter two phases aims to answer the **RQ 1**, if they are able to solve a programming task with the help of generative AI and whether they learn anything from it.
Three similar tasks were created, named A, B and C. To mitigate the influence of task order on the results and to avoid potential biases, the order of tasks, in which they were solved was randomized. This is to minimize the risk of systematic patterns emerging due to a specific order. The tasks were created within a single subject to assess whether participants had acquired knowledge by the time they reached the third task. Figure 3.2 show an example of the randomized order.
The tasks were set at 25 minutes each to ensure participants had enough time to engage with them. If the tasks were longer, the study would have approached two hours, and as mentioned in Section 3.1.1, most experiments did not last over two hours. We aimed to avoid a two-hour study because participating without a reward or incentive would be too demanding. Therefore, the tasks were not extended to prevent deterring participants due to the study's length.
While creating the tasks, the skill levels of bachelor's students were considered. By primarily focusing on bachelor's students, there is a greater chance for a uniform starting point for the majority of participants. The language that was chosen for the tasks was Java, which is one of the languages that the programs use in their curriculum. The tasks were also made in subjects that were not present in their education to introduce a new concept and leave room for learning. The subject that was chosen was web API programming, and an essential aspect of the task creation process involved ensuring that students could not obtain a fully completed solution solely by leveraging ChatGPT's capabilities to once again leave room for learning. To achieve this, newer or updated APIs, that ChatGPT 3.5 had yet to recognize or had updated information about, were used. At the time of the tasks creation, January 2024, ChatGPT's training data included information up to January 2022. This was constantly checked during the duration of the study, but no updates occurred during this time.

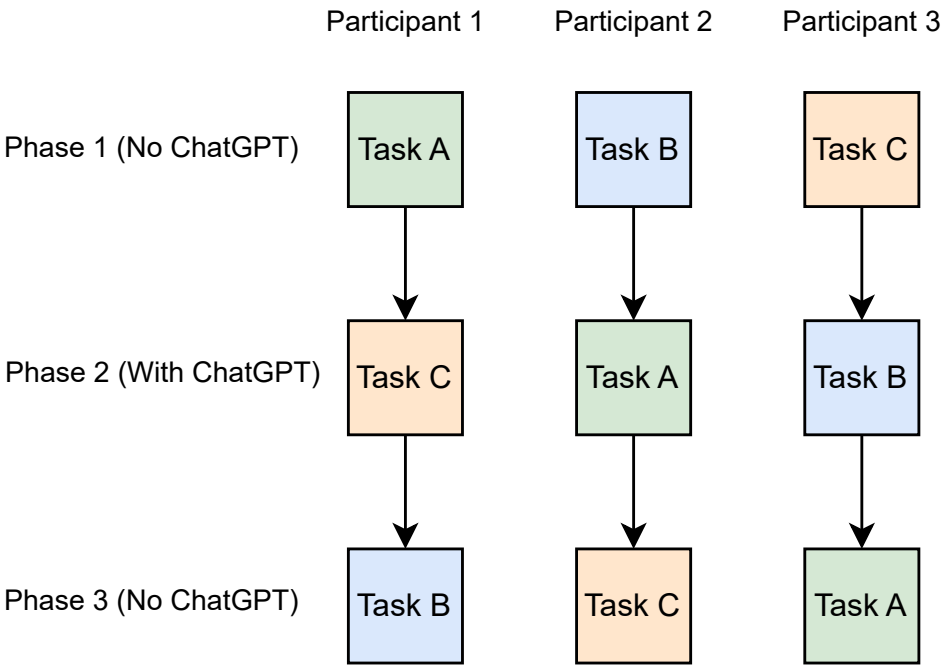


Figure 3.2: Example of task order

3.2.1 Pilot Study

Before conducting the experimental study it is crucial to run pilot studies beforehand since carrying out an experiment for the first time can expose weaknesses in the study design. This includes weaknesses such as confusing task description, unexpected decisions from the participants and difficulty level of the tasks as noted by Ko et al. [26]. The initial step of the pilot study was to identify and recruit participants. However, a crucial consideration was to ensure that the participants recruited for the pilot study would not overlap with those intended for the main study. One of the main reasons for this consideration stemmed from the awareness of us and our supervisor that there might be a potential challenge to recruit a sufficient amount of participants for the main study. Therefore, the participants in the pilot study consisted of master's students that are studying in the field of Computer Science. While there is a distinction between the two groups, the difference can be deemed as negligible for the purpose of this study. Furthermore, the choice for master's students can provide better insight on whether the programmings tasks are too challenging since they usually possess a higher level of programming knowledge and if the tasks turn out to be difficult for the master's students then there is a great chance that the tasks are too difficult for the bachelor's students.
The pilot study process mainly focused on having participants attempting some of the programming tasks within the defined time limit of 25 minutes per task, without incorporating the entire main study protocol. While we recognize the importance of maintaining a consistency between the pilot and main studies, we also acknowledged that the programming tasks would be the primary focus of the main study.

Participant	Program	Java	Web API
1	MPCSN (CTH)	+	-
2	MPCSN (CTH)	+	-
3	MPCSN (CTH)	+	-
4	MPALG (CTH)	+	-

Table 3.2: Participants' programs and whether they have experience with Java and Web API programming. Programs: MPCSN - Computer Systems and Networks Master's Programme, MPALG - Computer Science – Algorithms, Languages and Logic Master's Programme.

Therefore, our preeminent focus was to evaluate the tasks and the participants did not have to fill out an entry survey nor do an exit interview to keep the pilot study as short as possible, thus attracting more participants. The participants were instructed to solve the tasks without the help of ChatGPT to maximize the possibility of finding blockers and unforeseen actions.

After conducting the pilot study with 3 participants (the first three in Table 3.2), it was found that the tasks were clear and concise but quite challenging and participants got stuck on adding and working with dependencies, which was not intended. To address this issue, we came up with two potential solutions. The first option was to provide the participants which libraries and dependencies they were to utilize for the tasks. Meanwhile, the second option was to identify and pre-include all the libraries and dependencies that the participants could potentially use during their attempts. The potential downside for the second option was that participants might stumble upon libraries and dependencies that were not originally identified, thus bringing us back to the initial problem. Therefore, after some considerations, the second option was abandoned. This left us with the former option, which also came with the potential risk that the tasks might end up too simple. Hence, an additional pilot study was done to re-evaluate the tasks with the new changes (participant 4 in Table 3.2). The re-evaluation showed that the tasks did not end up to be too easy in the end. Additionally, during the task creation process, there were considerations to include tasks that required the utilization of an API key. However, it was ultimately decided that such task introduced unnecessary complexity and difficulty for the participants given their limited programming knowledge and time to complete the tasks, which was also mainly discovered through the pilot studies.

To evaluate the participants' performance on the tasks, we classified each task attempt into one of three categories: failed (-), partially solved (/), or solved (+). Partially solved in this context refers to participants that only managed to create an HTTP request to retrieve the data. The resulting task descriptions can be found in Appendix A.

3.3 Method

The resulting method was a controlled experiment with a within-subjects design and mixed methods analysis, which is described in Section 3.4.1. This was conducted to

determine the cause and effect of the following variables:

- Independent:
  - Use of generative AI (ChatGPT 3.5)
- Dependent:
  - Task completion (whether the task is solved)

To begin the study, the participants would fill out a consent form and an entry survey to record their programming skills and experiences, which took approximately five minutes. Before they began with the tasks, they were given a brief explanation of the skeleton code and how to run their code, and then the task solving would begin with a time limit of 25 minutes per task. In the first task, they were not allowed to use ChatGPT or any generative AI, but they were allowed to use the rest of the internet. In the second task, they were allowed to use ChatGPT. The last task was done without ChatGPT, the same way as the first task. After finishing each task, they were asked if they had learned anything during the task solving. After all the tasks were done, there would be an exit interview to allow them to speak about thoughts that they had during the study. They were also asked questions regarding their interaction with ChatGPT and if they had learned anything, which took approximately 10 minutes.

3.4 Data Analysis

Different data points were collected during the study. Firstly, an entry survey was conducted to record the participants' current level of programming knowledge. Then, prompts and answers exchanged with the AI to understand their strategies. The resulting code was also collected to determine whether the participants successfully solved the task. Observations were noted during the task-solving process to capture any relevant behaviors or interactions. Finally, an exit interview was conducted, during which participants were asked questions related to the research questions.

- Entry survey: A survey to record their current level of programming knowledge.
- Prompts and answers to/from AI: Analyze their strategies with ChatGPT
- Code: The resulting code and whether they solved the task or not
- Observations: Notes on observations during task solving
- Exit interview: An interview to give question related to the research questions

3.4.1 Mixed Methods Analysis

To analyze if a programmer can solve programming tasks using ChatGPT and learn from it, a mixed methods sequential explanatory analysis was employed. This approach involves collecting quantitative data first, followed by qualitative data within the same study. The quantitative data and their subsequent analysis provide a general understanding of the research problem, while the qualitative data refine and explain those statistical results, as explained by Ivankova et al. [27].
In the analysis of quantitative outcomes, whether they solved the task or not, we planned to use descriptive statistics to show patterns and trends within the data,

using different plots to showcase the data. However, given the low participation of our study, the usefulness of conducting descriptive statistics was limited. Using tables, key metrics were visualized to provide an overview of participants' performance when solving programming tasks with ChatGPT. Additionally, qualitative data, including exit interviews were incorporated to gain deeper insights into the participants' learnings and perceptions. The interviews provide contextual information about the participants experiences and their thought process while solving the tasks. By looking at their answer to questions such as "What did you learn?" and "How did you use ChatGPT to help you with the task?" we can uncover how participants engaged with the AI tool and the impact on their learning. This approach integrates both quantitative outcomes and qualitative experiences for a more comprehensive interpretation of the study's findings.

3.4.2 Thematic Analysis

To explore the strategies and approaches employed by participants when interacting with ChatGPT and solving programming tasks, a thematic analysis was conducted. This method, explained as "a way of systematically identifying, organizing, and gaining insight into patterns of meaning (themes) across a dataset" by Braun and Clarke [28], was employed to examine the collected data. The analysis made use of the different qualitative data, insights from the exit interview and the participants prompts to ChatGPT. To aid the analysis process, Taguette was used, which is a "free and open source qualitative research tool" as described by Rampin [29]. It allows for importing documents, highlighting and tagging.

Thematic analysis involved a systematic process of iterative coding, a process of labeling and categorizing qualitative data to identify recurring themes and patterns across the various data sources [30]. As highlighted by Williams and Moser [31], "A code in qualitative inquiry is most often a word or short phrase that symbolically assigns a summative, salient, essence-capturing, and/or evocative attribute for a portion of language-based or visual data". Additionally, the study employed an inductive approach due to the novelty of ChatGPT, where prior research is lacking. An inductive approach was deemed the most appropriate because it allows for the discovering of new theories directly from the data, rather than relying on preconceptions and pre-existing research. Therefore, this approach facilitates the emergence of codes and themes that are inherently data-driven [32].

The analysis began with semantic open coding, a starting phase to generate a set of broad codes to represent the high-level content of the data. This was to allow for quick identification of concepts and themes without interpretations. Afterwards, the process moved on to latent line by line coding, diving deeper into the data to find underlying meanings and connections. This method enabled a more thorough coding process and complex interpretation of the data, giving a better understanding of the data and finding insights that may have been overlooked in the initial coding. The coding process was conducted individually, with each of us analyzing the data independently to prevent biases from influencing each other's interpretations. Subsequently, individual codes were grouped into initial themes as patterns began to emerge from the data. Once these initial themes had taken shape, discussions were

held to review them. These discussions involved sharing insights, comparing interpretations, deriving themes that could be potential strategies.
To further revise the codes and see if the initial themes could develop into strategies, axial coding was employed. Additionally, the method was used to explore the possibility of identifying any other potential strategies. Williams and Moser [31] describe axial coding as a method used for refining, aligning, and categorizing themes. It is a usual step following open coding, where the focus shifts from identifying themes to refining them. The process involved going back to the codes to further revise them and to look for relationships between the codes, paying attention to what makes them connected or related. Codes were grouped into categories to capture main concepts in the data. Then the categories were organized based on the updated Strauss and Corbin axial coding paradigm model [33]. The model consists of three components, conditions, actions and consequences. Firstly, the conditions component refers to the various circumstances that lead to the emergence of an action. Secondly, the actions component, as the name implies, contain the different actions the participants took to solve the tasks. Lastly, the consequences component consists of different outcomes resulting from the actions taken.
After the categories had been designated a component in the paradigm model, potential strategies were theorized from the initial themes previously derived. Categories from the actions component were also used, since strategies often include various actions. With the potential strategies in mind, narratives were created based on the model to present how categories and codes connect to each other and to show how they can support and correspond to particular strategies. Following the narratives, we further explored the strategies to find deeper connections and gain a more nuanced understanding by analyzing how they interrelate with the other categories within our data. Finally, we compared these strategies by examining their conditions and consequences, identifying both differences and similarities.

4
Results

This section presents the results from the controlled experiment and thematic analysis.

4.1 ChatGPT's Impact on Task Success

In Table 4.1, the participants' results on each task are presented alongside the order in which they solved them. The table reveal that the majority of participants did not solve the first programming task, with Participant 2 being the exception. Particularly, Participant 2 had prior programming experience before university, as written in their entry survey where they wrote they had "at least 4 years of experience" in programming. Therefore, the Participant 2 was regarded as an outlier, who successfully completed all the tasks. While most participants managed to solve task 2 either partially or completely, a pattern was observed: those who successfully completed task 2 tended to partially solve the third task. While participants who were unable to fully complete the second task struggled to solve anything with task 3.

Participant	Task 1	Task 2	Task 3	Order
1	-	/	-	ABC
2	+	+	+	BCA
3	-	/	-	CAB
4	-	+	/	ABC
5	-	-	-	CBA
6	-	+	/	BAC
7	-	+	/	ACB

Table 4.1: Participants result on the experiment and the order in which the tasks were solved; (+, /, -) = (Solved, Partially Solved, Fail)

The exit interviews provide deeper insights into the outcomes of **RQ 1.1** and **RQ 1.2**. Across the board, participants express ChatGPT's helpfulness and its contribution to learning. Participants 4, 6 and 7 fully solved the task with ChatGPT and for task 3 they were able to make an HTTP request to fetch the data. Participant 6 and 7 express their takeaways:

> *"ChatGPT did most of the hard work, but I have learnt a lot more now. Especially on how to set up the request, client, and uri and how to send*

> *the request as well" — Participant 6 (Interview)*
>
> *"Maybe a bit more on how HTTP requests works. I told ChatGPT to do an HTTP request, but it also explained the code as well" — Participant 7 (Interview)*

The responses from participant 6 and 7 highlight the role of ChatGPT as both a problem-solving and educational tool. While acknowledging ChatGPT's contribution to completing the task, they also emphasize the learning's gained through the interaction. Participant 7 mentions that even if their intentions were just to do an HTTP request, ChatGPT provided not only the solution but also an explanation.
Participant 4 who also partially solved task 3 does not explicitly mention that they had learned anything but mentions:

> *"Googling for the first task was more keywords, but Google searches for the third task was a bit more how [they] used ChatGPT, using full sentences" — Participant 4 (Interview)*

Participant 4's observation regarding the shift in search behavior between tasks offers insight into the impact of utilizing ChatGPT. The progress from relying on keyword-based searches to formulating more comprehensive Google queries show a change in search strategy post-ChatGPT interaction. This shift show a more informed and strategic approach to the task following the usage of ChatGPT, that helped them make an HTTP request to fetch the data.
Participants 1, 3 and 5 were able to make some progress using ChatGPT compared to the first task, but they were only able to partially solve the task. Participant 1 and 3 mentioned that they learned a few things with participant 1 expressing that ChatGPT was helpful and:

> *"explained what needed to be done" — Participant 1 (Interview)*

The participant asked questions to ChatGPT during the task and afterwards said that:

> *"I have learned that I need to make a request to retrieve information from the API. Then I can use the information to iterate through specific values in order to update the desired result, if it satisfies a specific condition" — Participant 1 (Interview)*

showing a better understanding of the task. However, they did not learn enough to make an HTTP request to retrieve any data from the API. Participant 3 said after using ChatGPT:

> *"I learned that I can use HTTPRequest and builder or something close to that to import a web API " — Participant 3 (Interview)*

They have learned that making an HTTP request involves using a builder to con-

struct the request. However, they mentioned that this process allows them to "import a web API", which points out a misunderstanding. Web APIs are not imported like libraries but are accessed by sending HTTP requests to their endpoints. Therefore, their understanding of how web APIs work is incorrect. Participant 5 had trouble completing any of the tasks and did not manage to learn much either. They knew that they were supposed to get data using the API endpoint but did not get further than that expressing that:

> *"Needed way more time to accomplish something, especially since [they] didn't have any prerequisite knowledge" — Participant 5 (Interview)*

Participants 1, 3 and 5 had differences in their progress and learning, showing slight improvement in their knowledge. This was not enough and none of them were able to solve the third task, displaying a lack of knowledge and learning to solve the task. This suggests that their learning may have been more on conceptual understanding rather than practical application.

4.2 Thematic Analysis

The thematic codes are presented in this section, from the individual analysis to the combined analysis, where the themes are reviewed. Then the resulting themes and strategies are presented and explained.

4.2.1 Individual Analysis

Taguette was used to export all the codes as PDF files, which resulted in the appendices presented in this subsection. The appendices display from which participant the excerpt of data originates and which code or codes that excerpt is tagged with. The heading "Taguette highlights" implies codes from one analysis document, totaling eight, with four from each author.

Open coding

Appendix B.1 presents detailed results from the individual semantic open coding conducted on the interview/observational data. This resulted in:

- Researcher 1: 3 code labels with a total of 70 excerpts.
- Researcher 2: 10 code labels with 41 excerpts.

Appendix B.2 shows the results on the prompt data:

- Researcher 1: 6 code labels with 37 excerpts.
- Researcher 2: 9 code labels with 44 excerpts.

The detailed results are summarized in Table 4.2.

Researcher	Data	Code labels	Excerpts	Appx.
1	Interview and observations	3	70	A.1
2	Interview and observations	10	41	A.1
1	Prompts	6	37	A.2
2	Prompts	9	44	A.2

Table 4.2: Results from open coding

Line by line coding

Additionally, Appendices B.3 and B.4 showcase the results of individual latent line by line coding.

Appendix B.3 details the interview/observational data:

- Researcher 1: 41 code labels with 94 excerpts.
- Researcher 2: 21 code labels with 46 excerpts.

Appendix B.4 presents the prompt data:

- Researcher 1: 16 code labels with 61 excerpts.
- Researcher 2: 19 code labels with 56 excerpts.

The detailed results are summarized in Table 4.3.

Researcher	Data	Code labels	Excerpts	Appx.
1	Interview and observations	41	94	A.3
2	Interview and observations	21	46	A.3
1	Prompts	16	61	A.4
2	Prompts	19	56	A.4

Table 4.3: Results from line by line coding

Codebooks

Tables 4.4 through 4.7 present the four extracted codebooks from the individual thematic analysis. The codebooks consist of individually identified themes and a majority of all the codes, which are categorized accordingly to the themes. A majority of the themes and codes can be discovered across multiple participants, indicating that most of the participants did not solely rely on a singular strategy or approach.

During the open coding phase, there was more of a semantic-level focus, which resulted in codes that were broader and these were aimed to be more surface level codes. A few examples of these are “Strategy”, “Question”, “Request”, etc. Moreover, the line by line coding phase had a latent-level focus that resulted in more nuanced codes such as “Prompt to correct”, “Follow-up inquiry” and “Back and forth”. While identifying themes, it was particularly important to keep **RQ 2** in mind to ensure their relevance, this resulted in the identification of themes such as “Conversational” and “Independence”.

Theme	Codes
Conversational	Additional info prompts, Ask ChatGPT to clarify, Question to AI, Short prompts, Non-learning prompts
Task strategy/approach	Back and forth approach, Cautious, Task completion approach, Continuous interaction, Code adjusting, Copy answer, Didn't copy answer, Copy code, Command, Request, Inspection code, Time constraint, Error message debugging, ChatGPT approach on non-ChatGPT task, Task understanding, Good topic documentation
Knowledge and Learning	Crucial to have domain knowledge, Inexperienced with ChatGPT coding, Knowledge improvement after using ChatGPT, Learned to send request, Learning coding, Learning HTTPRequest, Prerequisite knowledge, Gain knowledge from previous task, Learn through ChatGPT explanation, Learning API, Faster start, AI literacy
Practical Considerations	Not job replacement, Easy to use, Perhaps too good, Positive ChatGPT experience, AI hallucinations, Situational preference, Good prompt, ChatGPT heavy work

Table 4.4: Extracted codebook #1 following line by line coding on interview/observational data

Theme	Codes
Understanding and Learning	Advice, Code inquiry, Assertion inquiry, Parsing Inquiry, HTTP inquiry, General question, Usage inquiry, For understanding
Interaction and Communication	Conversational, Follow-up inquiries, Prompt to correct
Task-Oriented and Goal-Driven	Focusing on finishing, General Request, Usage of task description
Independence	Advice, Manual Correction, Usage inquiry

Table 4.5: Extracted codebook #2 following line by line coding on interview/observational data

Theme	Codes
Debugging	Debugging
Clarification and Learning	Clarify code, Learning API data retrieval, Learning imports, Learning library, Learning web API
Request and Instruction	Request, Request code, Question — Simplify code option, Additional info prompts
Code Modification	Modification
Task Specificity	User specification, Specifying library, Retrieve data from API endpoint

Table 4.6: Extracted codebook #1 following line by line coding on prompt data

Theme	Codes
Exploration	Hypothetical, General Questions, Options, Usage inquiry
Clarifying and Correcting	Correct, Error, Additional Info, Response Questioning, Organization, Generate
Strict	Needing, Constraint
Lenient	Desire, Wanting, Request, Allow

Table 4.7: Extracted codebook #2 following line by line coding on prompt data

4.2.2 Combined Analysis

After reviewing and discussing the elicited codebooks, we found that despite an individual coding process, similar themes emerged independently in both of our codebooks. These common themes indicated a recurring pattern in the data and suggested shared insights on the participant's interaction with ChatGPT. The themes

"Understanding and Learning" and "Knowledge and Learning" were similar in name but also in that the codes often referred to participants knowledge and their efforts to learn, therefore these were merged into an initial strategy called "Knowledge and Learning". "Task-Oriented and Goal-Driven" was similar to "Task strategy/approach" where they both referred to participants constructing prompts mainly to complete the task and not to learn, thus these were merged into a second initial strategy called "Task Completion". Lastly, there were similarities between "Conversational" and "Interaction and Communication", that referred to participants that often went back-and-forth with the AI tool. As a result, these were merged into a third initial strategy called "Conversational". These merged themes, identified independently through our coding process, formed the three initial strategies, "Knowledge and Learning", "Task Completion" and "Conversational", which we aimed to validate through further analysis.

4.2.3 Axial Coding

In Figure 4.1, we present the results from the axial coding process, exploring relationships between categories of codes based on Strauss and Corbin axial coding paradigm [34]. The paradigm involved a structured method for connecting categories by focusing on conditions, actions and consequences.

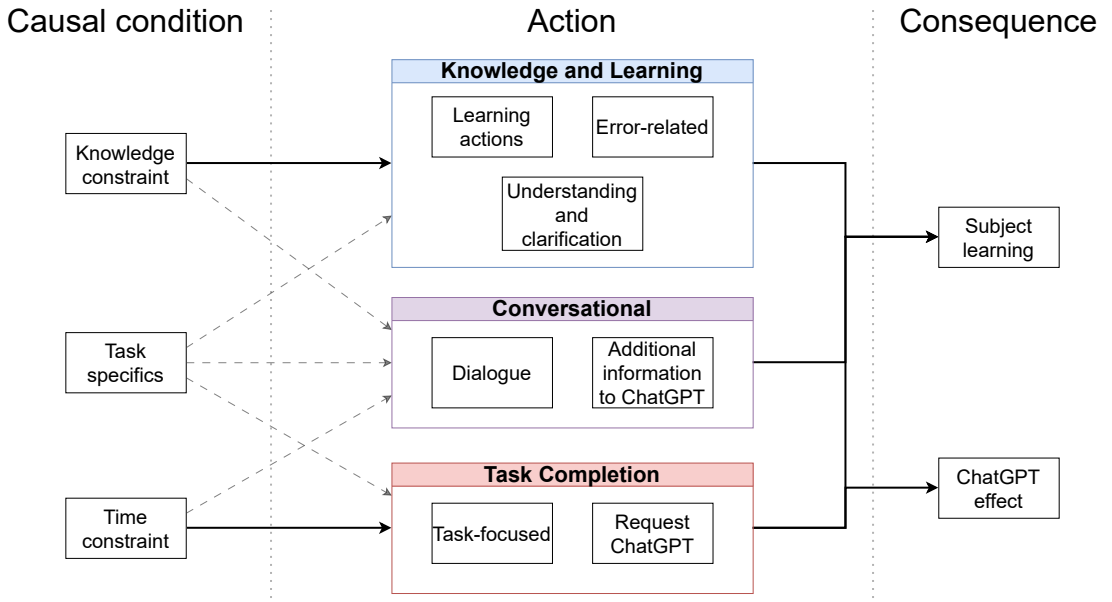


Figure 4.1: Result from the axial coding process, displaying categories based on Strauss and Corbins coding paradigm

The following content describe the different categories, highlighted with italics, and include excerpts. In some cases, the excerpts have been translated from Swedish to English. Then the relationships between the categories and how they can relate to different strategies and approaches that participants employ.

Conditions

One of the major conditions was the participant's prior programming skills and *Knowledge constraint*. All participants had at least one or two programming course, giving them foundational knowledge. However, the tasks were made in a subject that the participants' had not been exposed to yet in their education to allow learning during the process. Therefore, they were not familiar with concepts required for the task, such as HTTP requests and parsing JSON data. This influenced how they used ChatGPT's assistance.
Another condition was the *Time constraint* appointed on the participants. They were given a fixed amount of time to complete the programming tasks, which required them to manage their time while navigating the task using ChatGPT for assistance.
The *Task specifics* was also another condition. The instructions provided to the participants specified what they needed to do, such as fetching data and parsing the data to find a specific value. This description explained the requirements and steps necessary to complete the task, showing what was expected of them. Their understanding and usage of these instructions shaped how they approached the task and formulated their prompts to ChatGPT.

Actions

Various actions were seen as the participants used ChatGPT to solve the programming tasks. These actions include how participants interacted with ChatGPT, how they prompted the AI, and their approaches to understanding and applying the responses they received.
There were participants that did *Learning actions*. These include both the prompts participants used to acquire new knowledge or skills and their reflections during the interviews, where they expressed their approach. During the task, participants asked ChatGPT specific questions aimed at learning how to perform particular tasks or how to use specific libraries. Similarly, in the exit interviews, there were participants that mentioned how they used ChatGPT to ask questions to learn about specific things or how to use them. The examples show participants' efforts to use ChatGPT as a learning tool to gain more knowledge to solve the task:

> *"How do I create a program..." — Participant 1 (Prompt)*
>
> *"What does the code do?" — Participant 1 (Prompt)*
>
> *"How can I extract data..." — Participant 5 (Prompt)*
>
> *"How do I use the libraries..." — Participant 6 (Prompt)*

Closely related are the actions made for *Understanding and clarification*. While the learning actions focus more on learning how to do and use specific things, these actions are more towards gaining a general understanding of concepts. Attempts to ensure that the participant comprehend solutions and explanation to gain a better understanding of unfamiliar programming concepts are seen in some prompts:

"Prompts that gave answers to questions" — Participant 1 (Interview)

"What is this program doing?" — Participant 1 (Prompt)

"What is object mapper?" — Participant 5 (Prompt)

Another observed action was *Looking for options*, where participants sought advice on what to do or explored alternative solutions to their programming tasks. This involved asking ChatGPT for different ways to approach a problem or for clarification on potential methods they could use. Below are examples that indicate the interest in the range of possible approaches they could take to make decisions on implementing their code:

"What would that look like if the API endpoint is... " — Participant 3 (Prompt)

"What would that look like if each elixir has a name... " — Participant 5 (Prompt)

"Asked what [I] should do" — Participant 6 (Interview)

Contradictory to learning actions, *Task-focused* actions refers to users who engage with the AI primarily to finish the task rather than seeking for knowledge and understanding. These actions are generally focused on quickly obtaining a solution. As expressed by participant 4, disregarding learning anything and focusing mostly on finishing the task:

"Approach was just to finish the task in time" — Participant 4 (Interview)

"The way I asked it, I didn't learn anything" — Participant 4 (Interview)

Request ChatGPT to perform specific tasks was a common action observed among participants. This involved directly requesting ChatGPT to do various things for them, such as generating code. The nature of the requests varied, but they all shared the similarity of participants relying on ChatGPT to handle aspects of the programming tasks. There were statements that showed a direct approach to utilizing ChatGPT. Participants requested ChatGPT to generate complete solutions or to perform specific tasks, showing a tendency to assign parts of the problem-solving to the AI:

"Do this for me" — Participant 1 (Interview)

"Write me a java program... " — Participant 2 (Prompt)

"Retrieve data about games from this web api... " — Participant 4 (Prompt)

"Find which game... " — Participant 5 (Prompt)

"Asked it to do what I wanted" — Participant 7 (Interview)

Participants had different *Ways to express a need for something* through their

prompts. The choice of words showed the urgency of their requests. With prompts ranging from casual desires to more urgent or specific requirements:

"I want to..." — *Participant 1 (Prompt)*

"I need to retrieve data..." — *Participant 4 (Prompt)*

"the data should be saved..." — *Participant 6 (Prompt)*

Copy was the action where participants copied both code to be prompted but also code which ChatGPT had presented was also observed. This involved participants moving code snippets between their development environment and ChatGPT:

Copy code from vscode — *Participant 1 (Observation)*

Copy pasted the code — *Participant 1 (Observation)*

"Copy and paste a bit the answer from ChatGPT" — *Participant 3 (Interview)*

Threw in a whole error message — *Participant 6 (Observation)*

Participants also made actions where they added *Additional information to ChatGPT*, providing more context or constraints to receive more specific answers. Examples include specifying libraries, variables, or data structures, the additional information served to generate more relevant responses:

"These are the only imports I am using..." — *Participant 3 (Prompt)*

"...using the java.net.http..." — *Participant 4 (Prompt)*

"Using these variables" — *Participant 4 (Prompt)*

"The data I will receive are 'elixirs' which have the fields name, ingredients, ..." — *Participant 6 (Prompt)*

"In java using java.net.http subpackage" — *Participant 7 (Prompt)*

Code modification was an action observed among participants. This involved making changes to existing code or creating new versions to suit the task. Participants guided ChatGPT to provide corrections and refined their code to fit the task requirements. They not only relied on ChatGPT but also utilized their own skills to make necessary adjustments and improvements.

"Make another version of the program" — *Participant 1 (Prompt)*

Adjusting the code themselves — *Participant 3 (Observation)*

"Rewrite the code..." — *Participant 4 (Prompt)*

Tries to steer ChatGPT in the right direction — *Participant 6 (Observation)*

The *Dialogue* action refers to users that engaged with the AI in a conversational manner, characterized by shorter prompts, back-and-forth exchanges and follow-up inquiries. These actions aim to build upon previous responses from ChatGPT, which allows the user to explore the subject by iteratively asking for clarifications and questions.

> *"Gave a bit of info of what [I] wanted to do and added more info" — Participant 3 (Interview)*
>
> *Went back and forth a lot — Participant 6 (Observation)*
>
> *Asked question to ChatGPT about a response that he got — Participant 6 (Observation)*

Participants that used *Error-related* actions interacted with ChatGPT to diagnose and fix issues they stumbled upon. These action were made to identify causes of problems and finding solutions to continue their progress. Participants described their specific errors or warning they stumbled upon and used ChatGPT in understanding and fixing these issues.

> *"What is wrong here" — Participant 1 (Prompts)*
>
> *"I got a warning telling that following library cannot be resolved:" — Participant 5 (Prompts)*
>
> *"got this error, should i use a jsonarray instead of jsonobject?" — Participant 6 (Prompts)*

Consequences

The interactions with ChatGPT led to consequences that affected their learning and overall experience. These outcomes were different depending on what actions they took and the conditions they were in.
One category was named *ChatGPT effect* that explain the way users skills, behaviors and perceptions got affected after interacting with the AI. It was observed that Participant 2 was able to solve the first part of a task quicker with the help of ChatGPT, which show increased efficiency while using the AI. Additionally, Participant 1 stated that they felt the third task felt easier than the first one, thanks to using ChatGPT for the second task. It was also observed that they were able to start working on task three quicker than task one, which further show increased efficiency even post interaction and without ChatGPT. Furthermore, Participant 4, expressed that their approach on how to solve the task changed following the use of ChatGPT. As stated by Participant 7, they felt that solving the task with ChatGPT went perhaps a bit too fast, insinuating that ChatGPT might make one too dependent on the tool. However, they also expressed that the tool is great and recognizes ChatGPT's abilities when it comes to programming.

> *"Became easier after using ChatGPT" — Participant 1 (Interview)*

> *Started to work on parsing the data way quicker — Participant 2 (Observation)*
>
> *"3rd task was a bit more how [I] would use ChatGPT" — Participant 4 (Interview)*
>
> *"Maybe it went a bit too fast" — Participant 7 (Interview)*
>
> *"Great, avoided it for quite some time and didn't know how good it was at programming" — Participant 7 (Interview)*

The interactions with ChatGPT influenced the participants' learning experiences (*Subject learning*) in various ways. Some reported gaining a deeper understanding of programming concepts and techniques, such as performing HTTP requests, and some participants reflected on their learning strategies. For example, Participant 4 expressed the belief that asking for explanations from ChatGPT could have helped them learn, while Participant 5 shared the opinion that asking ChatGPT to explain code was helpful for understanding specific functions. Others mentioned acquiring knowledge about setting up HTTP requests and handling responses, as said by Participant 6. Participant 7 express that while their intention was to have ChatGPT perform an HTTP request, they also received an explanation of the solution, giving them an understanding beyond their original intentions.

> *"I have learned that I need to make a request to retrieve information from the API" — Participant 1 (Interview)*
>
> *"If I asked it to explain the answers than I would've learnt" — Participant 4 (Interview)*
>
> *"Think it helps a lot when it comes to learning specific functions, basically ask it to explain code" — Participant 5 (Interview)*
>
> *"...how to set up the request, client and uri, and how to send the request as well" — Participant 6 (Interview)*
>
> *"Asked ChatGPT to make an HTTP request, but it also answered what the code did as well" — Participant 7 (Interview)*

Possible Strategies

Since the action component is closely related to strategies, possible strategies were extracted directly from the action categories. A prominent action was *Learning actions*, which involved codes often referring to participants knowledge and their efforts to learn, which could be connected to the first initial strategy "Knowledge and Learning". Another action was *Task-focused*, which contained codes where participants constructed prompts mainly to complete a task and not to learn. This action could be connected to the second initial strategy "Task Completion". A third action was *Dialogue*, which included codes where participants interacted and went back-and-forth with the AI tool, thus being connected to the third initial strategy "Conversational".

Narratives

With the categories sorted into these paradigms, different narratives were formulated to explain the relationships between the categories. For instance, one narrative emerged from a single participant was:

> Narrative 1: Uncertain about how to fetch data from the API (**condition**), the participant asked the AI questions (**action**), which helped them learn that a request was needed to retrieve data from the API (**consequence**)

The participant, lacking knowledge on how to fetch data from the API, found themselves in a condition of uncertainty. This initial condition lead them to seek answers by using the AI's capabilities. The action taken was asking questions to the AI to get explanations and guidance. This action addressed their condition of uncertainty and helped them get a better understanding of their problem. As a result, they gained a better grasp of how to handle data fetching from an API, demonstrating a positive learning outcome. This narrative show the connection between the condition of lacking knowledge, the action of seeking answers, and the consequence of improved understanding. Another similar narrative was:

> Narrative 2: Due to a lack of knowledge (**condition**), the participant adopted a back-and-forth approach (**action**). They utilized the task description and asked the AI for advice **(action)**, as well as asked follow-up questions about the responses they received (**action**). The purpose was not to simply copy the answers but to understand the process (**action**). This approach helped them learn about the structure of setting up an HTTP request (**consequence**)

The participant found themselves in the condition of lacking knowledge. This condition led them to engage in a back-and-forth interaction with the AI. They used the task description as a reference point and sought advice. In addition, they asked follow-up questions to clarify and understand the responses provided by the AI. Instead of merely copying answers, the participant emphasized that their aim was to comprehend the responses. This approach helped them to learn the structure of setting up an HTTP request, displaying a positive learning outcome. Similarly to the previous narrative, this one highlight the relationships between the initial condition of lacking knowledge, the participant's active action to engage in a dialogue with ChatGPT, and the resulting understanding and skill development. Unlike the first narrative, which resulted in learning that an HTTP request was needed, the second narrative demonstrates more engagement by not only asking questions but also interacting with the AI. This resulted in a more thorough understanding of how to perform the HTTP request.
The narratives described above have similarities, pointing to the potential strategy "Knowledge and Learning". A strategy revolving around participants using ChatGPT as a learning tool. Another narrative is:

> Narrative 3: Due to the time limit (**condition**) for completing the task,

> Participant 4 approached using ChatGPT with the main goal of finishing the task on time (**action**). This approach led them to not learn anything (**consequence**).

The participant, constrained by a time limit, primarily focused on using ChatGPT effectively to complete the programming task. To achieve this, they used prompts that were intended for generating solutions quickly, rather than actually learning. As a result, the prompts were concise and very specific to the task at hand. However, the participant acknowledges that if they had asked ChatGPT to explain the answers, then they would have learned. Users that employ this approach do not dig deeper into the underlying concepts or ask for detailed explanations. Therefore, this narrative points to the potential strategy "Task Completion". A narrative that show a different perspective:

> Narrative 4: Having prior knowledge in the subject (**condition**), and understanding how to converse with ChatGPT to get what they need (**condition/action**) helped them get the whole code for retrieving the data (**consequence**)

In this narrative, the participant's knowledge of the subject and their understanding of how to use ChatGPT represented the initial conditions. These conditions provided a foundation for the participant to make informed actions. By using their subject knowledge, the participant communicated and conversed with ChatGPT effectively, providing relevant context to consequently obtain the complete code needed to retrieve data. The action of conversing with ChatGPT can be connected to the "Conversational" strategy. Another narrative that is connected to the "Conversational" strategy:

> Narrative 5: By using the task description (**condition**), the participant utilized shorter prompts and progressively added information (**action**). They communicated their need to know how something is done to ChatGPT (**action**) and made efforts to tweak the provided code themselves. This approach led to a positive ChatGPT experience, where they received useful hints on what to do (**consequence**) and regarded the AI as a helpful friend (**consequence**).

The participant's approach is characterized by their usage of the task description as a guiding condition, showing their awareness of the task requirements. Their action involve using shorter prompts and gradually adding relevant information, showing a communicative approach to using ChatGPT. They also prompted their requirements to the AI and demonstrated independence of refining the code themselves. The consequences of their actions result in both valuable hints and assistance from ChatGPT, as well as a positive perception of ChatGPT as a helpful friend, indicating a sense of reliance on the AI tool.

4.2.4 Strategies

This section digs deeper into the identified strategies and examines how other categories relates to them.

Knowledge and Learning

This strategy encompasses the participant's intentions of learning and gathering knowledge during the experiment. Participants engaged in understanding various aspects related to problem-solving, ranging from general clarification and more specific inquiries. As seen in Table 4.8, codes such as "For understanding" and "Ask ChatGPT to clarify" highlight the category *Understanding and clarification* where participants used ChatGPT to get clarification or overall understanding. One participant expressed:

> *"[They] did not copy answers, [ChatGPT] was mostly for [their] understanding" — Participant 6*

They used ChatGPT less as a problem-solving tool and more as a tool to gather deeper understanding of the task. Then there are more specific codes, such as the "Usage Inquiry" code. A code where participants prompt a desire to understand how to utilize specific tools or resources, such as asking:

> *"How can I extract data. . . " — Participant 5*
>
> *"How do I use the libraries?" — Participant 6*

These prompts aim to get answers that explain the functionality of the libraries, aiding the participant's learning process and helping them to effectively use the libraries. A majority of the participants stumbled upon errors and issues due to their inexperience regarding the subject. Prompt such as:

> *"What is wrong here" — Participant 1*

Underline the category *Error-related* action, in this case "Debugging". The participant's approach to fix an error was to ask what is wrong instead of asking it to fix the problem, which suggest that the participant wanted to learn.

Category	Code	Example	Data Source
Learning actions	Usage inquiry	"How can I extract data. . . "	Prompt
	Usage inquiry	"How do I use the libraries?"	Prompt
Error-related	Debugging	"What is wrong here?"	Prompt
Understanding and clarification	For understanding	"Did not copy answers, it was mostly for his understanding"	Interview
	Ask ChatGPT to clarify	"Asked ChatGPT to clarify"	Interview

Table 4.8: Knowledge and Learning Categories, Codes, examples and their respective data source

Task Completion

This theme pinpoints approaches participants used to accomplish the task, marked by goal-driven mindsets and prioritizing task completion. As opposed to "Knowledge and Learning" strategy where the intention is to gain understanding, this theme encloses actions towards solving the problem, such as utilizing the task description and prompting direct commands. Narrative 3 describes this strategy using examples found in Table 4.9 where the approach was:

> *"just to finish the task in time,"* noting that *"if [they] had asked it to explain the answers, then [they] would've learned more" — Participant 4*

indicating a more pragmatic approach focused on task completion over learning, but more importantly highlights the time constraint that motivated the strategy.
The *Time constraint* emerges as a factor motivating the usage of this strategy. Participants emphasized the limited time available, with one participant stating that there was:

> *"too little time" — Participant 3*

showing the time pressure set on the participants to complete the task within the time. The strategy was not solely driven by wanting immediate solutions, but was pushed by the time condition and by using a more *Task-focused* approach, participants aimed to tackle the time constraint
Other examples of the strategy include participants who:

> *"Asked [ChatGPT] to do what [they] wanted," providing commands such as "Make an HTTP request to fetch data. . . "- Participant 7*

In these instances, the participant use ChatGPT's capabilities to do specific tasks, such as retrieving data, by directly commanding it. This shows a *Task-focused* approach where participant focus more on completing the task. By offloading certain tasks to ChatGPT and relying on its speed and efficiency, they can address the time

constraint. This contrasts with the "Knowledge and Learning" strategy with codes such as "Usage Inquiry" where the participant prompt the need to understand how to use a web API before adding it into their solution.

Category	Code	Example	Data Source
Task-focused	Focusing on finishing	"Approach was just to finish the task in time"	Interview
Request ChatGPT	Command/Request	"Asked it to do what I wanted"	Interview
	Command/Request	"Make a http request to fetch data. . . "	Prompt
Constraint	Time constraint	"Too little time"	Interview
Task specifics	Usage of task-description	"Info from task description"	Interview

Table 4.9: Task Completion Categories, Codes, examples and their respective data source

Conversational

This theme highlights ChatGPT's conversational abilities, which are profoundly supported by its large training data and its ability to remember the context of the current conversation, giving it the capability to provide responses that are contextually coherent throughout the entire session. The strategy can be explained most broadly in the category *Dialogue*, which contain the code "Converse" and "Follow-up inquiries". Examples in Table 4.10, show the codes:

> *"converse with the AI" — Participant 2*
>
> *"Ask questions to ChatGPT about the response that [they] got" — Participant 7*

The first example showed the participant engaging in a back and forth communication. The second one had the participant referring back to ChatGPT's response
In a similar fashion, the category *Additional information to ChatGPT*, as the name suggests, focus on adding additional information and context to ChatGPT to get more specific results. In the category is the code "Corrective prompts" that focus on referring back to the previous response, but with the purpose of correcting a faulty response or providing extra information and constraints to get closer to the desired solution. An example show when a participant:

> *"Got a Python answer but corrected it with another prompt", the prompt being "In java using java.net.http" — Participant 7*

Here they specify the programming language and the library to be used for ChatGPT, pointing it in the right direction with an additional prompt.

The code "Short prompts" have the example where a participant express their approach where they:

> *"Give little info of what [they] want to do, and add a little bit more info"*
> *— Participant 3*

This approach was interpreted as a way to quickly interact with ChatGPT back and forth, where the smaller prompts indicate adding information as you go to further develop the answer.
The causal condition for this strategy is not as clear as the ones found in the other two strategies. In narrative 4, we see a participant motivation for using a conversational strategy, where they express that they know how to use ChatGPT and converse with it to get a desirable answer. For others, it was not a strategy used to tackle a constraint, but more as a way of interacting with ChatGPT.

Category	**Code**	**Example**	**Data Source**
Dialogue	Converse	"Converse with the AI"	Interview
	Follow-up Inquiries	"Asks questions to ChatGPT about the response that [they] got"	Observation
	Short prompts	"Give little info of what [they] want to do, and add a little bit more info"	Interview
Additional information to ChatGPT	Corrective prompt	"Got a python answer but corrected it with another prompt"	Interview
	Corrective prompt	"In java using java.net.http subpackage"	Prompt

Table 4.10: Conversational Categories, Codes, examples and their respective data source

4.2.5 Differences and Similarities

The "Knowledge and Learning" strategy is mostly motivated by the participant's knowledge constraints. Participants using this strategy use ChatGPT as a tool to enhance their understanding of the problem at hand and to acquire knowledge to tackle it. While the primary objective of the strategy is to gain knowledge, the search for knowledge is driven by the participant's need to comprehend the problem better and thereby solve it successfully.
In contrast, the "Task Completion" strategy is often motivated by the time constraint. Here, participants prioritize completing the task as fast as possible over acquiring new knowledge or understanding. Unlike the "Knowledge and Learning" strategy, which emphasize learning and comprehension, the "Task Completion" strategy is solely focused on finishing the task within the time constraint, without necessarily understanding or acquiring knowledge about the task.

The “Conversational” strategy, however, differs in its motivation. Unlike the other strategies, it lacks a clear causal condition for its use. Instead, it is used as a means to interact with ChatGPT, providing participants with a tool to engage in a conversation. The strategy can complement other approaches, such as the “Knowledge and Learning” strategy, by having a dialogue to gain knowledge while conversing with ChatGPT. This is also a reason as to how the “Conversational” strategy can be similar to the other strategies. As an example, participants do follow-up inquiries about the previous response they got from ChatGPT, which is both “Conversational” and “Knowledge and Learning”. What is more to the strategy is that it is focused on a conversational interaction with ChatGPT, this sets it apart from other strategies, particularly “Task Completion,” where participants primarily prompt requests and commands to ChatGPT, often disregarding its conversational abilities. Instead, the “Conversational” strategy prioritizes back-and-forth dialogue. While using the strategy “Task Completion” to come up with quick and efficient solutions, participants can end up in a scenario where they become overly reliant on the tool. As previously presented, participant 7 expressed that the AI might be too good, implying that one could perhaps become overly reliant on the AI tool or hinder development. Therefore, employing the “Knowledge and Learning” strategy over “Task Completion” is a more sustainable approach. As participant 6 conveyed, they did not copy the answers to allow more room for understanding, thus acting as a safeguard against becoming overly reliant on ChatGPT.

Unlike the strategies “Knowledge and Learning” and “Task Completion”, “Conversational” does not have any clear consequences. The reason being that the consequences of using a “Conversational” strategy often refers to the outcome of the former two strategies, depending on which one was used in combination with it.

While the “Task Completion” strategy disregard learning, it can also result in the same learning as the “Knowledge and Learning” strategy. Due to the ChatGPT’s irregular nature, it might provide an explanation to the provided solution, even if the prompt did not ask for it, which could lead to both of the strategies resulting in the same learning outcomes.

5
Discussion

This section discusses the results from the controlled experiment, including the strategies employed by participants. Lastly, limitations and threats to the validity are addressed.

5.1 Research Question 1

Research question 1 asks whether *"a person with limited programming experience can successfully solve a coding task with the assistance of AI"*. The results indicate that with AI assistance, individuals with limited programming experience can successfully solve a coding task, or at least partially solve it, performing better than they would without using ChatGPT. This improvement can be seen in the outcomes of Task 1 and 2.
The purpose of Task 1 was to assess participants' prior knowledge in programming and in the specific subject. This baseline would help differentiate whether the participants' success in the next task was due to their own skills or ChatGPT's capabilities. The results from Task 1 show that the majority of participants failed to solve the task, suggesting a lack of knowledge needed to complete it independently. This observation indicated that for the next task, participants would likely rely heavily on ChatGPT to succeed, providing a basis for assessing how ChatGPT influenced their completion of Task 2.
In Task 2, participants 1, 3 and 5 were able to partially solve it by making an HTTP request, while participants 4, 6 and 7 were able to fully complete the task by going further and finding the correct answer. The outcomes display an improvement compared to the first task and show an impact of ChatGPT that enabled them to overcome initial difficulties and make progress completing the coding tasks, showing its potential to aid individuals with limited programming experience.
Research question 1.1 and **1.2** build on the first research question. **RQ 1.1** explores the knowledge participants acquire when they successfully solve a task with ChatGPT's assistance. We found that ChatGPT helped them learn more about making an HTTP request, demonstrating its ability to teach new concepts. **RQ 1.2** investigates what participants learn if they fail to solve the task, even with ChatGPT's help. In these cases, participants still acquired some knowledge, but it was more general and conceptual rather than practical. This was observed in Task 3 and the interviews where participants were specifically asked what they learned.
In Task 3 we see the same divide in participants where participants 1, 3, and 5 failed the third task and participants 4, 6, and 7 did better and partially solved the task.

The participants that fared better expressed that using ChatGPT helped them learn more about making an HTTP request, indicating ChatGPT's ability to teach new concepts. Our study finds that AI does facilitate learning, and this finding is aligned with Popovici's study [17] where they spoke on the potential benefit AI has on learning. However, the knowledge gained is only on general concepts and not practical. The other participants expressed a need for more time to digest and learn, as well as to write the code. They mentioned that they learned more about web APIs, but in Task 3 we could see that they struggled to apply the knowledge practically. This showed that their learning was primarily on general concepts and understanding rather than on practical, applicable skills. One participant mentioned what they had learned after Task 2, but it turned out to be a misunderstanding regarding web APIs. This misunderstanding might have been due to time constraints, where they were unable to process the information provided by ChatGPT, leading to difficulties in Task 3. Our observations of these limitations are similar to the findings discussed in a study by Yilmaz and Karaoglan Yilmaz [14], where they investigate students' opinions after being able to use ChatGPT for assignments. While Yilmaz and Karaoglan Yilmaz obtained insights from students' experiences, our study directly observes these shortcomings. Specifically, participants in our study did not acquire practical skills and experienced misunderstandings, highlighting potential detriments to the development of critical thinking skills caused by fast responses from an AI, as suggested by Yilmaz and Karaoglan Yilmaz.

We see that participant 4, 6 and 7 did better overall and upon further investigation, it was observed that they had gotten a slight idea of what needed to be done from Task 1, even if they were not able to accomplish anything. This prior knowledge gave them an idea of what to look for when using ChatGPT, which may have influenced their success in Task 2. We addressed this potential vulnerability in the study by using exit interviews and having participants document what they learned after each task. This approach helped us account for learning from Task 1 in our analysis. While this makes it harder for us to draw a conclusion about the impact that ChatGPT had on their results, it is worth to note that this aligns closely with the null hypothesis and our initial expectations. We expected a lean towards the null hypothesis, where there would be no difference in the ability of beginner programmers to solve coding tasks with or without AI. This was based on the premise that you need to write a good prompt for a good answer. This could explain why these participants fully solved Task 2 compared to others or why others had trouble solving the task. Participants with prior understanding from Task 1 could produce informed prompts, resulting in more relevant responses from ChatGPT. Moreover, this understanding enabled them to engage more effectively with ChatGPT, ultimately leading to better performance in Task 3 as well. In contrast, participants without this knowledge spent more time figuring out basic steps, which affected their performance in both Task 2 and Task 3. These two types of interactions are similar to the findings of Barke et al. [35], who identified two modes of interaction with AI assistants. The first mode, called Acceleration mode, is where "the programmer already knows what they want to do next, and Copilot helps them get there quicker." The second mode, called Exploration mode, is where "the programmer is not sure how to proceed and uses Copilot to explore their options or

get a starting point for the solution."
Additionally, the results align with the findings of Jing et al. [13], who explored the impact of Matplotlib, a comprehensive library for creating visualizations in Python, and programming knowledge base on the effectiveness of using ChatGPT to solve programming tasks. They observed that participants with a solid foundation in Python could provide ChatGPT with clear and precise instructions based on their own Python knowledge, thus enabling them to complete programming tasks more effectively. In contrast, participants who lacked a programming knowledge base found it challenging to transform programming tasks into a series of questions suitable for natural language processing using ChatGPT. While Jing et al. [13] found that specialized knowledge in Matplotlib increased effectiveness, it was not as impactful as a general programming knowledge base. This suggests that while specialized knowledge can enhance performance, a broader knowledge base offers more significant benefits. Similarly, our study speculates that more knowledge in the subject offers better results, reinforcing the idea that prior knowledge, whether in programming or the subject at hand, increases performance when using ChatGPT.

5.1.1 Implications

Our findings indicate that ChatGPT can assist inexperienced programmers to varying degrees, making programming more accessible to beginners. By aiding with initial steps, ChatGPT helps beginners feel encouraged to continue learning as they achieve small progressions with AI assistance. Moreover, the effectiveness of ChatGPT is increased when users have prior knowledge. Therefore, users might consider an initial learning or training phase before using AI tools to maximize their benefits. ChatGPT can in short time offer quick solutions leading to conceptual understanding, which is helpful. However, overreliance on quick immediate answers can reduce the time dedicated to problem-solving, possibly hindering developing practical skills. This highlight the balance needed between using ChatGPT's accessibility and developing basic knowledge and critical thinking skills necessary for learning and skill development.

5.2 Research Question 2

Research question 2 and **2.1** asks "what are the strategies and approaches employed by beginner programmers when utilizing generative AI assistance for programming tasks?" and "What are the benefits and challenges associated with this usage?" Three major strategies emerged from the analysis.
The first strategy, "Knowledge and Learning", focuses on acquiring knowledge by using prompts that ask questions to learn specific actions and gain a broader understanding. This strategy facilitates learning, but if used hurriedly, might not result in significant learning and requires time to be effective. Another challenge is that if you have no prior knowledge about the topic, it can be difficult to identify what is relevant in ChatGPT's responses.
The second strategy, "Task Completion," prioritizes completing tasks, often using prompts that command ChatGPT to perform most of the work, with little focus

on learning. While useful for solving tasks quickly, this approach can lead to an overreliance on ChatGPT and limited learning. However, the strategy is beneficial when users know what they are working on, do not need to learn, and can create informed prompts.

The third strategy, "Conversational," emphasizes interacting with ChatGPT in a back-and-forth conversation, leveraging ChatGPT's conversational abilities. This creates a more positive and natural interaction, similar to talking with a friend, unlike traditional methods where you have to search for information on your own. This strategy can be combined with other approaches, such as using the conversation for knowledge and learning.

In a similar study by Khojah et al. [36], the authors discuss different purposes for using ChatGPT and how they can affect a user's experience. The purposes are in line with our identified strategies. The "Training" purpose in the paper aligns with the "Knowledge and Learning" strategy. Both involve using ChatGPT to acquire new knowledge and gain broader understanding by asking detailed questions. Similarly, the "Artifact manipulation" which is used in "a goal-oriented manner with the expectation that ChatGPT will produce or modify a concrete solution", resembling our "Task completion" strategy. The distinction we have is similar to that in the paper, where one is more goal-oriented and focused on finding a solution, while the other emphasize learning rather than directly solving a specific problem.

The "Knowledge and Learning" was mainly motivated by the participant's lack of knowledge, while the "Task Completion" was mostly driven by the time constraint. However, it is important to note that all participants were influenced by both constraints. Most participants struggled with the first task, indicating their lack of knowledge, and all participants had the same limited time for each task. Therefore, the decisions they made could be due to a combination of these factors, making the strategies more related. For example, participants might use the "Knowledge and Learning" strategy for task completion, as their ultimate goal was to finish the task. In a similar way the "Task Completion" might be used because of their lack of knowledge. What makes them different is the participant's main focus and which condition they want to address the most.

By using the "Knowledge and Learning", participants would seek to understand how things work by asking questions and looking for explanations. This strategy can help facilitate learning but can also fall short if it is used hurriedly. In a calm environment without time pressure, unlike our experimental setting, this strategy could lead to deeper and more practical learning. Such an environment would allow participants to fully digest the information, reducing the risk of misunderstanding due to time pressure. Additionally, using this strategy might also require knowing what to look for, where users might need sufficient knowledge or understanding, to ask the right questions. Without a basic understanding of the subject, it can be challenging to differentiate relevant information from unrelated details that ChatGPT might provide. This supports the null hypothesis, which assumes that AI has no effect on beginners' ability to solve coding tasks, and aligns with the idea that prior knowledge is necessary to formulate informed prompts or questions to AI.

The "Task Completion" strategy can be useful for completing tasks and assist in learning. Due to its irregular nature, it can also provide explanation of the provided

solution even if it was not asked for. However, users might ignore these explanations and incorporate the solutions without much thought, especially if their main focus is simply to get the solution. This strategy can also lead to overreliance on ChatGPT, a concern mentioned by both our participants and those in other studies, such as Yilmaz and Karaoglan Yilmaz [14]. The participants in that study acknowledged the benefits of using AI tools but also recognized the disadvantages, such as hindering the development of critical thinking skills. Like the "Knowledge and Learning", "Task Completion" also requires some initial knowledge to formulate good prompts. Similarly, Khojah et al. [36] note that individuals often struggle to provide sufficient context to ChatGPT because they are unsure of what context is needed, which sometimes led to frustration with AI's perceived shortcomings. This further highlights the importance of understanding what information to include in prompts. Therefore, this strategy is best used when the user already has a good understanding of what they want to achieve and does not need to learn new concepts, allowing them to focus on completing their task.

The strategies used by participants when interacting with ChatGPT also resemble the various prompt patterns identified in prompt engineering, as presented by White et al. [18].

In the prompt category of prompt improvement, which aims to enhance the quality of both input and output, the "Questions refinement" pattern stands out. This pattern ensures that the AI suggests improved alternatives to the user's initial question. This is similar to the "Conversational" strategy observed in our study, where participants engage in a back-and-forth interaction with ChatGPT. They formulate their next prompt based on the previous response received, aiming to guide the AI towards generating an improved alternative response.

In the same category is "Alternative approaches" that requires the LLM to suggest alternative methods to accomplish a given task. This can look like the "Task Completion" strategy, where a user seeks quick alternative solutions to finish the task.

The participants employed different strategies when interacting with ChatGPT. But most did not use the many different categories and patterns found in prompt engineering, despite their purpose in helping interaction with AI systems. There are potential benefits of familiarizing oneself with prompt engineering principles when using AI. By understanding these patterns, users can form their prompt more successfully to get more desired responses from the AI.

Similar to our previous observation, where prior knowledge about a subject helps users in formulating informed prompts, familiarity with prompt engineering principles can better one's ability to create prompts that generate accurate and relevant responses from AI models. We think that learning prompt engineering can be beneficial for people intending to use ChatGPT.

5.3 Threats to Validity

In this section, we discuss the potential threats that could have impacted the validity of our results and the strategies employed to mitigate them.

5.3.1 Internal Validity

As noted by Jhangiani et al. [37], one of the main issues with a study that uses a within-subjects design is the possibility of carryover effect, which is a type of order effect. Carryover effect occurs when participants' are exposed to information in a task that may affect their performance in a later task. An example of this is the practice effect, when participants' perform better on a task simply because they were exposed to a similar task previously. Thus, leading to improved performances due to familiarity rather than, in this case, learning from using ChatGPT. In contrast, there is the fatigue effect, which is the opposite of the practice effect. Participants' could become tired or bored, thus affecting their performance on later tasks. A solution to order effects is counterbalancing, which is when the experimental conditions are tested in different orders. In the case of this study, all possible orders with the three conditions A, B and C were used, which is also called complete counterbalancing.
Observing participants as they solve their tasks is another limitation. Solving problems while being watched may be uncomfortable, impacting their performance. Embarrassment and pressure can skew the results [38].
The entry survey could pose a risk for stereotype threat. The phenomenon in this case refers to where we ask the participants' to fill out their programming knowledge and experiences before attempting to solve the tasks. This can negatively affect the participants' confidence and performance due to reminding them about their limited experience [39].
All the experiments did not run in parallel, which raises the issue that earlier participants could inform later participants about the study design and tasks used during the experiment. This was mostly tackled by asking the participants in the end to not share any information about the experiment to anyone they knew was going to participate in the study.
If participants had prior knowledge of prompt engineering, it could influence their performance, particularly in comparison to participants who were unfamiliar with the concept. This difference could introduce a bias, as those with prompt engineering experience might perform better due to their familiarity with structuring their prompts. The entry survey only asked about prior use of ChatGPT and did not specifically ask about experience with prompt engineering. As a result, this could pose as a threat to our findings.
Another potential threat to the validity of our study is the variation in cognitive abilities among participants. Some participants may have stronger problem-solving skills, which could lead to better performance regardless of their prior use of ChatGPT or programming knowledge. This variation in cognitive abilities can introduce a bias, as it may not be the study's variables but rather individual differences in cognitive skills that account for performance differences. Measuring cognitive abilities accurately is challenging, so without accounting for these differences, this can pose as a threat to the validity.

5.3.2 External Validity

A small sample size limits the generalizability of the findings, and the results may not be representative of the broader population of students learning programming.

Additionally, given that the study is supposed to be conducted within a certain time frame, a delimitation such as no extra follow-up assessment after a longer time period is not feasible, which further decreases the generalizability.
Generative AI models like ChatGPT are constantly evolving, with each version becoming more intelligent. This combined with the controlled experiment being conducted in a laboratory setting can lead to an ecological validity threat. As described by Schmukler [40], ecological threat is "Whether or not one can generalize from observed behavior in the laboratory to natural behavior in the world". In this case, the results from the experiments could differ from real world results due to the constant development of ChatGPT. At the time of writing (May 2024), after finalizing the experiments, a new version of ChatGPT has been released. OpenAI's newest flagship model ChatGPT-4o is available for free, however, with limited number of prompts for free tier users. Despite this, it can pose significant challenges for studies that rely on its outputs, such as this one, where the goal with the tasks was to involve APIs that ChatGPT had yet recognized. If this model had been released in the middle of the study before we could finalize all the experiments, the consistency and reliability of the results could have been affected. Out of curiosity, we used ChatGPT-4o's free prompts to see whether it would recognize the APIs used in the study. It was able to browse the web and obtain up-to-date information and generated an answer that would inform the user what the API was and how to use it. Therefore, had this version been released during the experiments, it is likely that the tasks would have become significantly easier with the assistance of ChatGPT.

6
Conclusion

ChatGPT can assist individuals with limited programming experience in solving parts of a task. Our findings indicate that having some prior knowledge or a basic understanding of the subject enhances the likelihood of success, as shown by participants who used insights from Task 1 to perform better in the next tasks. This observation aligns closely with our hypothesis.

Regarding learning, ChatGPT proves effective in providing an initial and overarching understanding of a problem. However, it falls short in enabling participants to practically apply this knowledge to solve tasks, primarily due to time constraints. Therefore, while ChatGPT can ease the learning process and task completion, it is most beneficial when users already have a foundational understanding of the subject. Having prior knowledge allows users to effectively formulate prompts and differentiate relevant information from irrelevant ones. Additionally, when users have time to digest the information provided by ChatGPT, they can engage more with the AI, leading to deeper learning outcomes.

The thematic analysis highlighted "Knowledge and Learning", "Task Completion" and "Conversational" as the three main strategies that beginner programmers employ while using ChatGPT. The conditions played a major role in determining the strategies chosen by the participants, which included the time and knowledge constraint. The "Knowledge and Learning" strategy is driven by the need to understand, meanwhile "Task Completion" is driven by the time pressure to finish the tasks. However, all participants were affected by both factors, which suggests that a participant's decision to pick a specific strategy depends on their current situation and how comfortable they are. The "Conversational" is different from the other strategies. It is more a way of interacting with ChatGPT, where the benefits lie in the natural and conversational interactions. Users can ask questions and receive answers, resulting in a positive attitude towards AI usage. This stands in contrast to the traditional way of looking for information that lacks this conversational aspect. Users must seek information alone, without opportunity for dialogue and discussion. This may not provide the same level of engagement or satisfaction as a conversational interaction.

The analysis highlights the potential benefits of learning prompt engineering and patterns to improve the quality of responses. When users combine their prior knowledge with the prompt engineering principles to formulate better and informed prompts, they can engage in deeper interactions with ChatGPT. This ultimately leads to more favorable outcomes and responses. Therefore, it is advised for programmers, especially beginners to gain some initial domain knowledge before relying entirely on ChatGPT to solve coding tasks for you.

6.1 Future Work

As discussed in Section 5, there are several areas where future work on the impact of generative AI on learning programming could build upon and improve the groundwork of this study. Firstly, a larger sample size would greatly enhance the stability of the quantitative analysis by capturing a wider range of characteristics and variations that could be present in the population. Secondly, extending the study duration would allow for follow-up assessments with the participants. This would provide more insight about the long-term effects of using generative AI on learning programming. Lastly, future work could benefit from comparing different AI models such as ChatGPT, Gemini and Llama to provide a broader perspective on how different models influence programming learning outcomes. By examining the strengths and weaknesses of each model, we could gain more insights into the factors that lead to learning experiences. Thus, enhancing the generalizability of the findings.

Bibliography

[1] K. Martineau, "What is generative ai?" IBM, 2023. [Online]. Available: https://research.ibm.com/blog/what-is-generative-AI

[2] S. Peng, E. Kalliamvakou, P. Cihon, and M. Demirer, "The impact of ai on developer productivity: Evidence from github copilot," 2023.

[3] J. Qadir, "Engineering education in the era of chatgpt: Promise and pitfalls of generative ai for education," 12 2022.

[4] I. Education, "Ai code-generation software: What it is and how it works," september 2023, accessed on 2023-12-10. [Online]. Available: https://www.ibm.com/blog/ai-code-generation/

[5] GuardRails, "Ai-assisted coding: A double-edged sword," december 2023, accessed on 2023-12-10. [Online]. Available: https://www.guardrails.io/blog/ai-assisted-coding-a-double-edged-sword

[6] R. Michel-Villarreal, E. Vilalta-Perdomo, D. E. Salinas-Navarro, R. Thierry-Aguilera, and F. S. Gerardou, "Challenges and opportunities of generative ai for higher education as explained by chatgpt," *Education Sciences*, vol. 13, no. 9, 2023. [Online]. Available: https://www.mdpi.com/2227-7102/13/9/856

[7] S. Sok and K. Heng, "Opportunities, challenges, and strategies for using chatgpt in higher education: A literature review," *Journal of Digital Educational Technology*, vol. 4, no. 1, Jan 2024.

[8] J. Meyer, R. Urbanowicz, P. Martin, K. O'Connor, R. Li, P.-C. Peng, T. Bright, N. Tatonetti, K. Won, G. Gonzalez, and J. Moore, "Chatgpt and large language models in academia: opportunities and challenges," *BioData Mining*, vol. 16, 07 2023.

[9] N. Humble, J. Boustedt, H. Holmgren, M. Goran, S. Seipel, and A.-S. Östberg, "The consequences of chatgpt for programming education: Cheating or ai-enhanced learning?" 12 2023.

[10] D. Baidoo-Anu and L. Ansah, "Education in the era of generative artificial intelligence (ai): Understanding the potential benefits of chatgpt in promoting teaching and learning," *Journal of AI*, vol. 7, 03 2023.

[11] E. Alasadi and R. Baiz, "Generative ai in education and research: Opportunities, concerns, and solutions," *Journal of Chemical Education*, vol. 100, 07 2023.

[12] M. Rahman and Y. Watanobe, "Chatgpt for education and research: Opportunities, threats, and strategies," *Applied Sciences*, vol. 13, p. 5783, 05 2023.

[13] Y. Jing, H. Wang, X. Chen, and C. Wang, "What factors will affect the effectiveness of using chatgpt to solve programming problems? a quasi-experimental study," *Humanities and Social Sciences Communications*, vol. 11, 02 2024.

[14] R. Yilmaz and F. G. Karaoglan Yilmaz, "Augmented intelligence in programming learning: Examining student views on the use of chatgpt for programming learning," *Computers in Human Behavior: Artificial Humans*, vol. 1, p. 7, 07 2023.

[15] A. Tlili, B. Shehata, M. Adarkwah, A. Bozkurt, D. Hickey, R. Huang, and B. Agyemang, "What if the devil is my guardian angel: Chatgpt as a case study of using chatbots in education," *Smart Learning Environments*, vol. 15, pp. 1–24, 02 2023.

[16] R. Yılmaz and F. G. Karaoğlan Yılmaz, "The effect of generative artificial intelligence (ai)-based tool use on students' computational thinking skills, programming self-efficacy and motivation," *Computers and Education: Artificial Intelligence*, vol. 4, p. 100147, 06 2023.

[17] M.-D. Popovici, "Chatgpt in the classroom. exploring its potential and limitations in a functional programming course," *International Journal of Human–Computer Interaction*, p. 1–12, Oct. 2023. [Online]. Available: http://dx.doi.org/10.1080/10447318.2023.2269006

[18] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with chatgpt," 2023.

[19] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023.

[20] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. Le, and E. Chi, "Least-to-most prompting enables complex reasoning in large language models," 2023.

[21] Y. Jing, H. Wang, X. Chen, and C. Wang, "What factors will affect the effectiveness of using ChatGPT to solve programming problems? A quasi-experimental study," *Humanities social sciences communications*, vol. 11, no. 1, 2 2024. [Online]. Available: https://www.nature.com/articles/s41599-024-02751-w

[22] F. Fittkau, "Controlled experiments in software engineering," 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:63720009

[23] D. Sjoeberg, J. Hannay, O. Hansen, V. Kampenes, A. Karahasanovic, N.-K. Liborg, and A. Rekdal, "A survey of controlled experiments in software engineering," *IEEE Transactions on Software Engineering*, vol. 31, no. 9, pp. 733–753, 2005.

[24] A. Blandford, A. Cox, and P. Cairns, "Controlled experiments," *Blandford, A. and Cox, A.L. and Cairns, P.A. (2008) Controlled experiments. In: Cairns, P.A. and Cox, A.L., (eds.) Research Methods for Human Computer Interaction. Cambridge University Press, Cambridge, UK, pp. 1-16. ISBN 9780521870122*, 01 2008.

[25] R. Budiu, "Between-subjects vs. within-subjects study design," *Nielsen Norman Group*, 07 2023. [Online]. Available: Between-Subjectsvs. Within-SubjectsStudyDesign

[26] A. J. Ko, T. D. LaToza, and M. M. Burnett, "A practical guide to controlled experiments of software engineering tools with human participants," *Empirical*

Software Engineering, vol. 20, no. 1, p. 110–141, Sep 2013.

[27] N. Ivankova, J. Creswell, and S. Stick, "Using mixed-methods sequential explanatory design: From theory to practice," *Field Methods*, vol. 18, pp. 3–20, 02 2006.

[28] V. Braun and V. Clarke, *Thematic analysis.*, 01 2012, pp. 57–71.

[29] Remi Rampin, "Taguette," https://gitlab.com/remram44/taguette, 2023, accessed: April 29, 2024.

[30] D. S. Cruzes and T. Dyba, "Recommended steps for thematic synthesis in software engineering," in *2011 International Symposium on Empirical Software Engineering and Measurement*, 2011, pp. 275–284.

[31] M. Williams and T. Moser, "The art of coding and thematic exploration in qualitative research," *International Management Review*, vol. 15, p. 45, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:198662452

[32] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative research in psychology*, vol. 3, no. 2, pp. 77–101, 1 2006. [Online]. Available: https://doi.org/10.1191/1478088706qp063oa

[33] G. Kaiser and N. Presmeg, Eds., *ICME-13 Monographs*, ser. ICME-13 Monographs. Cham: Springer Nature, 2019, read page 123. [Online]. Available: http://library.oapen.org/handle/20.500.12657/23016

[34] A. Strauss and J. M. Corbin, *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*. Sage Publications, Inc., 1990.

[35] S. Barke, M. B. James, and N. Polikarpova, "Grounded copilot: How programmers interact with code-generating models," 2022.

[36] R. Khojah, M. Mohamad, P. Leitner, and F. G. de Oliveira Neto, "Beyond code generation: An observational study of chatgpt usage in software engineering practice," 2024.

[37] R. S. Jhangiani, I. A. Chiang, C. Cuttler, and D. C. Leighton, *Research Methods in Psychology*. Pressbooks, 2019.

[38] R. Spano, "Potential sources of observer bias in police observational data," *Social Science Research*, vol. 34, no. 3, pp. 591–617, 2005.

[39] C. M. Steele and J. Aronson, "Stereotype threat and the intellectual test performance of african americans," *Journal of Personality and Social Psychology*, vol. 69, no. 5, pp. 797–811, 1995.

[40] M. A. Schmuckler, "What is ecological validity? a dimensional analysis," *Infancy*, vol. 2, no. 4, pp. 419–436, 2001.

A

Appendix — Task Descriptions

These are the full task descriptions used during the experiments. Whatever, task that was given during phase 2 (with ChatGPT) the text "but not AI" was erased from the description.

Task A (Wizard world)
You are to create a program that interacts with a web API to retrieve information about Elixirs from a Wizard World!

- This is the specified API endpoint to fetch elixir information. https://wizard-world-api.herokuapp.com/Elixirs.
- The data consists of a collection of elixirs. Each elixir has:
  - A name
  - Effects
  - Potential side effects
  - Difficulty of preparation
  - Ingredients
- Your task is to **find the elixir that has the most ingredients**. Please use the predefined variables when creating your solution.
- You are not expected to know everything from the start. Use the internet (but not AI) to solve the task.
- The solution does not require you to add other dependencies
  - To retrieve the data, use the built-in java.net.http sub-package
  - To handle the data, use the google gson library

Task B (PRC Exam)
You are to create a program that interacts with a web API to retrieve information about Exams in the Philippines!

- This is the specified API endpoint to fetch exam information. https://api.whenisthenextboardexam.com/exams/2023.
- The data consists of a collection of exams. Each exam has:
  - A name/profession
  - A Type
  - Year
  - Dates
  - Places/Locations for the exam
  - End date for applications
  - Date when results are announced

- Your task is to **find the exam which has the most dates**. Please use the predefined variables when creating your solution.
- You are not expected to know everything from the start. Use the internet (but not AI) to solve the task.
- The solution does not require you to add other dependencies
  - To retrieve the data, use the built-in java.net.http sub-package
  - To handle the data, use the google gson library

Task C (CheapShark)

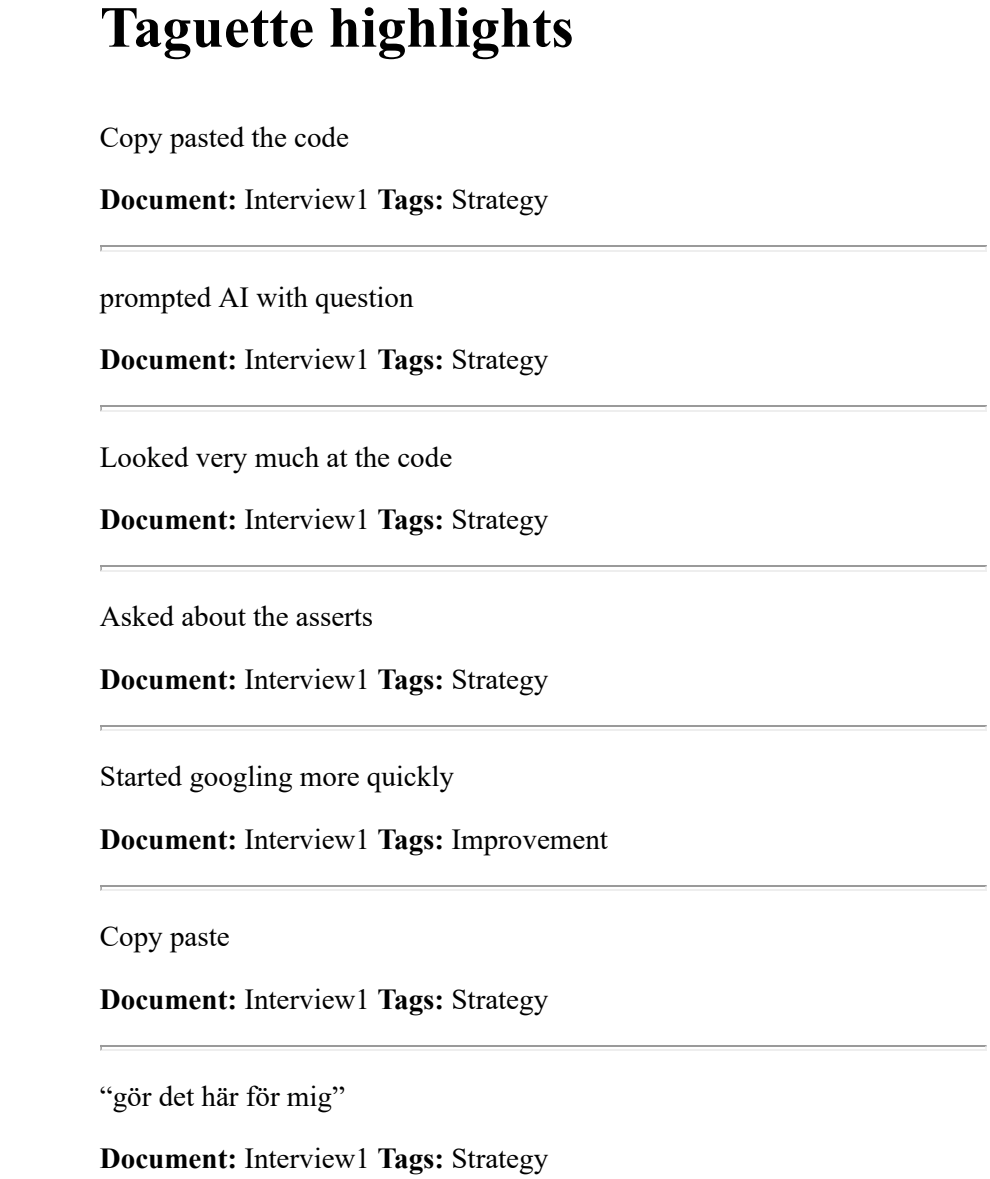
You are to create a program that interacts with a web API to retrieve information about prices for digital PC Games.

- This is the specified API endpoint to fetch elixir information. https://www.cheapshark.com/api/1.0/deals?storeID=1&upperPrice=15.
- The API endpoint consists of a collection of games. Each game has a certain amount of data information.
- Your task is to **find the game which has the highest "savings" and "title" associated with that specific game**. Please use the predefined variables when creating your solution
- You are not expected to know everything from the start. Use the internet (but not AI) to solve the task
- The solution does not require you to add other dependencies
  - To retrieve the data, use the built-in java.net.http sub-package
  - To handle the data, use the google gson library

B

Appendix — Taguette

B.1 Open coding on interview/observational data



Taguette highlights

Copy pasted the code

Document: Interview1 **Tags:** Strategy

---

prompted AI with question

Document: Interview1 **Tags:** Strategy

Looked very much at the code

Document: Interview1 **Tags:** Strategy

Asked about the asserts

Document: Interview1 **Tags:** Strategy

Started googling more quickly

Document: Interview1 **Tags:** Improvement

Copy paste

Document: Interview1 **Tags:** Strategy

“gör det här för mig”

Document: Interview1 **Tags:** Strategy

Figure B.1: Example codes #1 following open coding on interview/observational data

Taguette highlights

Ingen aning hur man hämtar data från API

Document: Interview1 **Tags:** Knowledge

---

prompted AI with question

Document: Interview1 **Tags:** Question

Hämta och använda datan

Document: Interview1 **Tags:** Difficulty

“gör det här för mig”

Document: Interview1 **Tags:** Request

Frågade: Vad gör koden?

Document: Interview1 **Tags:** Question

---

prompts som gav svar på frågor

Document: Interview1 **Tags:** Question

Task 2/C (10:41-10:46)
Read the task and then used ChatGPT on how to retrieve data with the java net http sub package

Figure B.2: Example codes #2 following open coding on interview/observational data

B.2 Open coding on prompt data

Taguette highlights

how do i cretate a program that finds the highest game which has highest savings and the title associated with that specific game

Document: Participant 1 Prompts.txt **Tags:** Implementation/Requ

---

i want to ge tthe info from the api endpoint and use the predifine v to create the solution

Document: Participant 1 Prompts.txt **Tags:** User specification

---

you are adding more http imports than the ones in my code,. why

Document: Participant 1 Prompts.txt **Tags:** Clarification/Learnin

---

what is wrong here

Document: Participant 1 Prompts.txt **Tags:** Debugging

---

can you keep the assert results in the end o the program

Document: Participant 1 Prompts.txt **Tags:** Organize code, User specification

---

but i want to keep them as they are i dont want to put them in a m

Document: Participant 1 Prompts.txt **Tags:** User specification

---

Figure B.3: Example codes #1 following open coding on prompt data

Taguette highlights

how do i cretate a program

Document: Participant 1 Prompts.txt **Tags:** Question

---

i want to ge tthe info from the api endpoint

Document: Participant 1 Prompts.txt **Tags:** Desire

---

you are adding more http imports than the ones in my code

Document: Participant 1 Prompts.txt **Tags:** interesting

---

why

Document: Participant 1 Prompts.txt **Tags:** Question

---

what is wrong here

Document: Participant 1 Prompts.txt **Tags:** Question

---

can you keep the assert

Document: Participant 1 Prompts.txt **Tags:** Request

---

i want to keep them as they are i dont want to put them in a method

Document: Participant 1 Prompts.txt **Tags:** Correct, Desire

Figure B.4: Example codes #2 following open coding on prompt data

B.3 Line by line coding on interview/observational data

Taguette highlights

copy code

Document: Interview1 **Tags:** Copy code

Copy pasted the code

Document: Interview1 **Tags:** Copy code

---

prompted AI with question

Document: Interview1 **Tags:** Question to AI

Looked very much at the code

Document: Interview1 **Tags:** Inspecting code

Started googling more quickly

Document: Interview1 **Tags:** Faster start

---

chatGPT förklarade vad man behövde göra

Document: Interview1 **Tags:** Positive ChatGPT experience

Copy paste

Document: Interview1 **Tags:** Copy code

Figure B.5: Example codes #1 following line by line coding on interview/observational data

Taguette highlights

Ingen aning hur man hämtar data från API

Document: Interview1 **Tags:** Knowledge

---

copy code from vscode

Document: Interview1 **Tags:** Copy

Copy pasted the code

Document: Interview1 **Tags:** Copy

---

prompted AI with question

Document: Interview1 **Tags:** General question

Asked about the asserts

Document: Interview1 **Tags:** Assertion inquiry

Hämta och använda datan

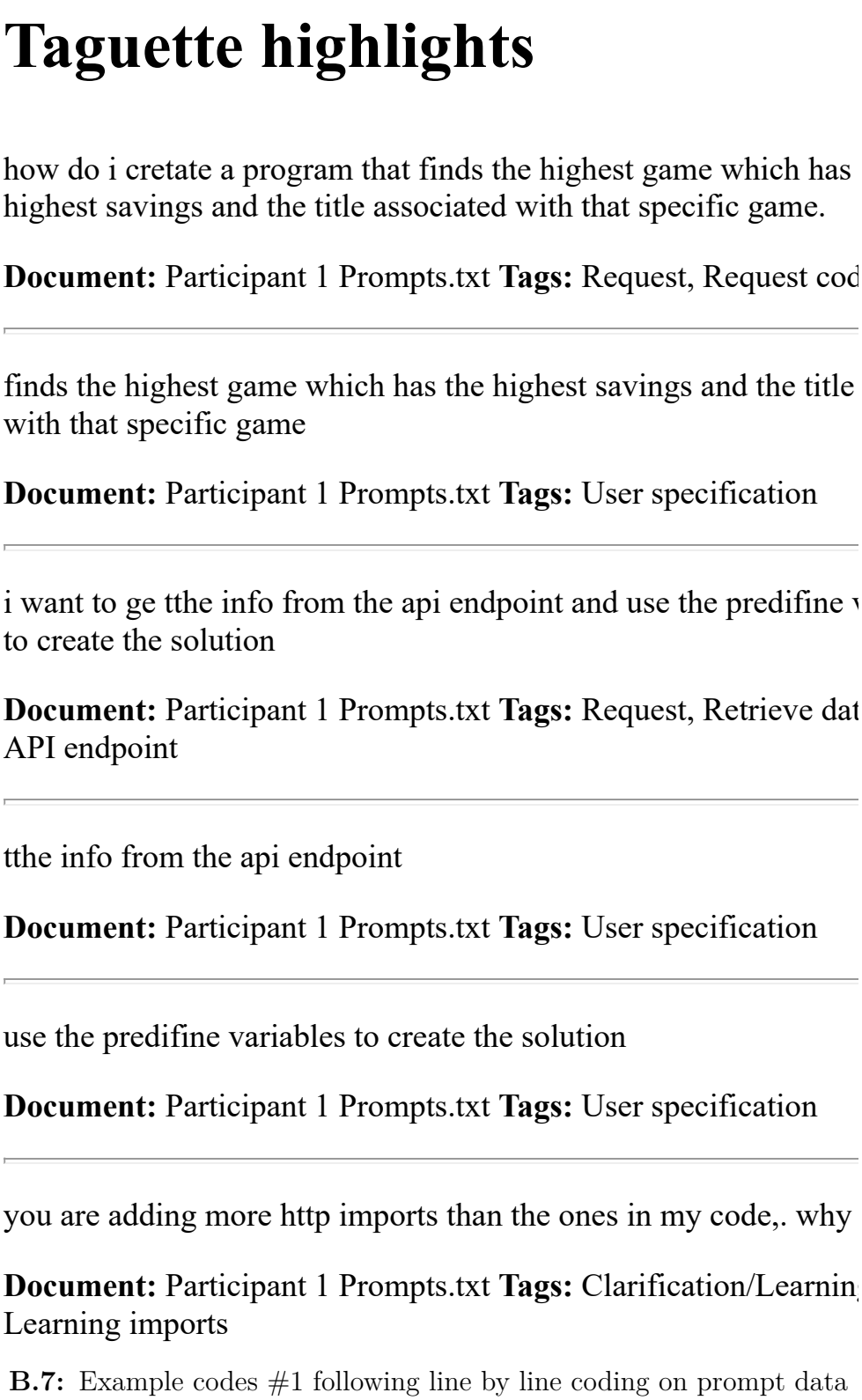
Document: Interview1 **Tags:** Difficulty

“gör det här för mig”

Document: Interview1 **Tags:** General Request

Figure B.6: Example codes #2 following line by line coding on interview/observational data

B.4 Line by line coding on prompt data



Taguette highlights

how do i cretate a program that finds the highest game which has highest savings and the title associated with that specific game.

Document: Participant 1 Prompts.txt **Tags:** Request, Request cod

---

finds the highest game which has the highest savings and the title with that specific game

Document: Participant 1 Prompts.txt **Tags:** User specification

---

i want to ge tthe info from the api endpoint and use the predifine v to create the solution

Document: Participant 1 Prompts.txt **Tags:** Request, Retrieve dat API endpoint

---

tthe info from the api endpoint

Document: Participant 1 Prompts.txt **Tags:** User specification

---

use the predifine variables to create the solution

Document: Participant 1 Prompts.txt **Tags:** User specification

---

you are adding more http imports than the ones in my code,. why

Document: Participant 1 Prompts.txt **Tags:** Clarification/Learnin Learning imports

Figure B.7: Example codes #1 following line by line coding on prompt data

Taguette highlights

how do i cretate a program

Document: Participant 1 Prompts.txt **Tags:** Question, Generate

---

that finds

Document: Participant 1 Prompts.txt **Tags:** Retrieval

---

i want to ge tthe info from the api endpoint

Document: Participant 1 Prompts.txt **Tags:** Wanting

---

use the predifine variables

Document: Participant 1 Prompts.txt **Tags:** Constraint

---

to create the solution

Document: Participant 1 Prompts.txt **Tags:** Generate

---

you are adding more http imports than the ones in my code

Document: Participant 1 Prompts.txt **Tags:** interesting

---

why

Document: Participant 1 Prompts.txt **Tags:** Response questioning

Figure B.8: Example codes #2 following line by line coding on prompt data