

Master's Thesis

# ON BASELINES FOR PROGRAM COMPREHENSION STUDIES WITH FMRI

ANNABELLE BERGUM

March 31, 2021

Advisor:

Norman Peitek M.Sc. Chair of Software Engineering

Examiners:

Prof. Dr. Sven Apel

Chair of Software Engineering

Prof. Dr. Antonio Krüger German Research Center for Artificial Intelligence

Chair of Software Engineering  
Saarland Informatics Campus  
Saarland University





## **Erklärung**

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

## **Statement**

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis

## **Einverständniserklärung**

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

## **Declaration of Consent**

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, \_\_\_\_\_  
(Datum/Date)

\_\_\_\_\_  
(Unterschrift/Signature)



## ABSTRACT

---

In the research community on understanding program comprehension using a neuroscientific approach, *functional Magnetic Resonance Imaging* (fMRI) studies are used to measure cognitive load in program comprehension, but with very different baselines. fMRI is a non-invasive technique with a high spatial resolution which parts of the brain are used. Depending on the baseline used, the results of the studies vary and are thus difficult to compare. So that a study which uses reading as a baseline might conclude that mathematics is the crucial factor in program comprehension while another study might use mathematics as a baseline and will conclude that program comprehension is based on reading. In this thesis, we present tasks from the domains of language, mathematics, and mental load that we examine for the capability to serve as baselines for program comprehension studies. In the first step we present a literature review. In the second step we present an online study, which tests, if the tasks are understandable and how long they take to complete. Based on our results, we outline a future fMRI study to further evaluate the selected tasks. Further the proposed fMRI study using these tasks, to prove if they can serve as a baseline, can be conducted. With a proper standardized baseline, the results of individual studies can be better compared against each other. Furthermore, if the activated areas of basic tasks involved in program comprehension such as reading or looking at the screen are already subtracted, the nuances in different code snippet, such as iterative vs. recursive, can be better highlighted in the brain activity.



## ACKNOWLEDGMENTS

---

First, I thank Prof. Dr. Sven Apel for this fascinating topic. I had never heard of this interdisciplinary research, between neurosciences and computer science, however after a relatively short time I realized that this is exactly what I want to do.

Additionally, I would like to thank Norman Peitek M.Sc. for his good guidance into scientific working and writing.

Furthermore, I appreciate Prof. Dr. Krüger for taking the time to review this thesis.

Thank you also for the support, suggestions and constructive criticism on my experiment, the Software Engineering chair and the Neuroage research group.

I also thank my friends and family who helped me hunting for typos and my husband for always having a hot cup of tea on my writing table.

Last but not least, thank you to everyone who participated in my online study and thus generated the statistical data for this thesis.





# CONTENTS

---

1	INTRODUCTION	1
1.1	Goal of this Thesis . . . . .	1
1.2	Overview . . . . .	2
2	BACKGROUND	3
2.1	fMRI . . . . .	3
2.2	Brodmann Areas . . . . .	4
2.3	Cognitive Load . . . . .	4
2.4	Program Comprehension . . . . .	6
3	RELATED WORK	7
3.1	Baselines . . . . .	7
3.2	fMRI Studies on Language Processing . . . . .	10
3.3	fMRI Studies on Mathematics . . . . .	11
3.4	fMRI Studies on Mental Load . . . . .	13
3.5	fMRI Studies on Program Comprehension . . . . .	17
3.6	Summary of the Related Work . . . . .	21
4	SUGGESTED TASKS	25
4.1	Language-Oriented Tasks . . . . .	25
4.2	Mathematical Tasks . . . . .	26
4.3	Mental Load-Oriented Tasks . . . . .	26
4.4	Program Comprehension Tasks . . . . .	29
4.5	Validity . . . . .	30
5	EXPERIMENT	31
5.1	Prestudy for the fMRI Study (Main Study in this Work) . . . . .	31
5.1.1	Demographic Questions . . . . .	31
5.1.2	Online Survey Tool . . . . .	32
5.1.3	Setup . . . . .	32
5.1.4	Screenshots of the online survey . . . . .	34
5.2	Research Questions . . . . .	36
6	EVALUATION	37
6.1	Evaluation Pipeline . . . . .	37
6.1.1	Technical . . . . .	37
6.1.2	Clean Data . . . . .	37
6.1.3	Participant Demographics . . . . .	42
6.1.4	Response Times . . . . .	45
6.1.5	Correctness Rate . . . . .	48
6.1.6	Original vs. New Program Comprehension Tasks . . . . .	51
6.1.7	Performance Prediction with Elastic Net . . . . .	53
6.1.8	Research Questions . . . . .	56
6.2	Threats to Validity . . . . .	57
7	FMRI STUDY DESIGN	59
7.1	Requirements for an fMRI Experiment Design . . . . .	59

7.2	Task Selection . . . . .	59
7.3	Derived fMRI Study Design . . . . .	65
7.4	fMRI Study Onsite . . . . .	66
8	CONCLUDING REMARKS . . . . .	67
8.1	Conclusion . . . . .	67
8.2	Future Work . . . . .	67
A	APPENDIX . . . . .	69
	BIBLIOGRAPHY . . . . .	71

## LIST OF FIGURES

---

Figure 2.1	An <i>fMRI</i> device . . . . .	3
Figure 2.2	An example of <i>Raven Progressive Matrices</i> ( <i>RPM</i> ) . . . . .	5
Figure 2.3	An example of <i>Mill Hill Vocabulary</i> ( <i>MHV</i> ) . . . . .	5
Figure 3.1	An example of cross fixation . . . . .	7
Figure 3.2	An example of visual noise . . . . .	8
Figure 3.3	An example of a match task . . . . .	14
Figure 3.4	An example of a figural task . . . . .	14
Figure 3.5	An example of an analytic task . . . . .	15
Figure 3.6	An example of cube figures . . . . .	15
Figure 3.7	An example of the task used in the study by Krueger . . . . .	18
Figure 3.8	An example of the task used in the study by Huang . . . . .	19
Figure 3.9	An example of the "fake code" . . . . .	20
Figure 3.10	Overview of activated <i>Brodmann Area</i> ( <i>BA</i> )s in language-oriented studies and program comprehension . . . . .	22
Figure 3.11	Overview of activated <i>BAs</i> in math-oriented studies and program comprehension . . . . .	22
Figure 3.12	Overview of activated <i>BAs</i> in mental load studies and program comprehension . . . . .	23
Figure 4.1	Sandia Matrix Generation Software . . . . .	27
Figure 4.2	Examples of our <i>RPM</i> tasks . . . . .	28
Figure 4.3	Structure of a task . . . . .	30
Figure 5.1	Structure of a block . . . . .	32
Figure 5.2	Structure of the online study . . . . .	33
Figure 5.3	Online Survey: program comprehension . . . . .	34
Figure 5.4	Online Survey: math comprehension answer . . . . .	35
Figure 6.1	Response statistic of the online survey . . . . .	37
Figure 6.2	Boxplot explanation . . . . .	39
Figure 6.3	Correctness rate among participants . . . . .	40
Figure 6.4	An example on comprehension time among tasks . . . . .	41
Figure 6.5	Outlier rate among participants . . . . .	41
Figure 6.6	Visualization of the clean data process . . . . .	42
Figure 6.7	Overview over the age and years at university, programming and <i>Java</i> programming . . . . .	43
Figure 6.8	Overview over the qualifications of our participants . . . . .	44
Figure 6.9	Comprehension time per task type . . . . .	45
Figure 6.10	Answer time per task type . . . . .	45
Figure 6.11	Significance Test: Answer Time . . . . .	46
Figure 6.12	Significance Test: Comprehension Time . . . . .	47
Figure 6.13	Examples of comprehension time distribution in a single task . . . . .	47
Figure 6.14	Correctness rate per task type . . . . .	48

Figure 6.15	Significance Test: Correctness Rate . . . . .	48
Figure 6.16	Answer option distribution (Math) . . . . .	49
Figure 6.17	Answer option distribution (Language) . . . . .	49
Figure 6.18	Answer option distribution (Program Comprehension) . . . . .	50
Figure 6.19	Answer option distribution (RPM) . . . . .	50
Figure 6.20	Comparison original and new code snippets . . . . .	51
Figure 6.21	Significance Test: Comprehension Time and Correctness Rate . . . . .	53
Figure 7.1	Data of the math-oriented tasks sorted according to the criteria . . . . .	62
Figure 7.2	Data of the language-oriented tasks sorted according to the criteria . . . . .	63
Figure 7.3	Data of the RPM sorted according to the criteria . . . . .	63
Figure 7.4	Data of the program comprehension tasks sorted according to the criteria . . . . .	64
Figure 7.5	Timetable of the fMRI study . . . . .	65

## LIST OF TABLES

---

Table 3.1	An overview of the presented studies and review papers on language processing . . . . .	10
Table 3.2	An overview of the presented studies on mathematics . . . . .	11
Table 3.3	An example of the word riddle and the corresponding equation . . . . .	12
Table 3.4	An overview of the presented studies and review papers on mental load . . . . .	13
Table 3.5	An overview of the presented studies on program comprehension . . . . .	17
Table 6.1	Comprehension time and correctness rate of all program comprehension tasks . . . . .	52
Table 6.2	Results of Elastic Net: Language, Math and RPM used to predict program comprehension time . . . . .	54
Table 6.3	Results of Elastic Net: demographics used to predict program comprehension . . . . .	55

## LISTINGS

---

Listing 3.1	Two example sentences from the paper by Newman et al. [25] . . . . .	10
Listing 4.1	Example of a language-oriented task . . . . .	25
Listing 4.2	Example of a mathematical task generated by our python script . . . . .	26
Listing 4.3	Example of a program comprehension task . . . . .	29

## ACRONYMS

---

BA	<i>Brodmann Area</i>
BOLD	<i>Blood Oxygenation Level Dependent</i>
CSV	<i>Comma-Separated Values</i>
EEG	<i>Electroencephalography</i>
fMRI	<i>functional Magnetic Resonance Imaging</i>
fNIRS	<i>functional Near-Infrared Spectroscopy</i>
IDE	<i>Integrated Development Environment</i>
MHV	<i>Mill Hill Vocabulary</i>
MNI	<i>Montreal Neurological Institute</i>
MTL	<i>Medial Temporal Lobe</i>
RPM	<i>Raven Progressive Matrices</i>
TUT	<i>task-unrelated thoughts</i>



## INTRODUCTION

---

### 1.1 GOAL OF THIS THESIS

For decades, researchers have been studying the human brain. Especially in the field of language processing, many studies have been conducted. *Broca* and *Wernicke areas* have been discovered and named after their discoverers. Since then, the techniques that allow us to look inside the human head have developed significantly. Therefore, instead of waiting for injuries or creating them artificially, we can now take non-invasive images of the brain. One of these techniques is called *functional Magnetic Resonance Imaging (fMRI)*. It allows us to observe the brain functions while the participant does certain tasks. With this invention, it is now possible to do a lot more studies that aim to explore the functioning of the brain, as we have less ethical restrictions and it is easier to find volunteers.

However, not only the technology, but also the research questions have changed. We now aim at understanding the cognitive processes. With [fMRI](#), we can make them measurable. Furthermore, we are interested in comprehending how different kinds of problems affect thinking and how various people think in different ways (e.g., with increasing experience in the domain of the problem). Recently, mathematical skills and programming skills are also studied and investigated. All of this is only possible because we now have [fMRI](#). For example, the mental effort of the programmers can be observed with the help of [fMRI](#). It objectively indicates how intensively the programmers have to think during a task. The cognitive load is then measured for different types of code presentation. Thus, we can determine how to design code that generates the lowest cognitive load. From these measurements, we can then derive guidelines for how code should be displayed. In addition, the tools for source code generation can be improved so that programming becomes easier. This would increase the productivity of programmers. Therefore, many areas will benefit from this work in the long term.

Nevertheless, there are still many open questions with using [fMRI](#) to study programmers. We will present one of them in this work, namely the problem of a suitable baseline and explore possible solutions. A baseline would ideally be the state without brain waves which are triggered by thoughts. This would purely be the brain waves that are necessary to keep the body alive. Unfortunately, some studies have shown that higher levels of brain activation occurred at times during the resting phase of the study than during the time when the participants were working on their tasks [4]. Many participants also reported thinking about other things during the rest period (e.g., reflective thinking such as "What will I eat for lunch"). Consequently, the rest condition cannot be the solution for the baseline, since it is difficult to stop the wandering of thoughts.

One approach is not to consider the brain without thoughts as a baseline, but to present the participant a specifically designed baseline task that differs from the test task in exactly one aspect. The baseline task should take approximately the same amount of time to complete. In essence, during analysis of an *fMRI* study the brain effort during the task is subtracted by the brain effort during the baseline. The resulting difference in brain areas and their brain activation strength can then be localized in the brain. A good baseline makes it clearer to identify the differences between, for example, different types of presentation of the source code, if both tasks are compared to the baseline before being evaluated against each other. The aim of this work is to investigate which baseline tasks can be used to find the areas in the brain that are involved during program comprehension.

## 1.2 OVERVIEW

At the beginning, in Chapter 2, the terms used, such as *fMRI* and program comprehension, are introduced. Then, in Related Work, Chapter 3, the work in the areas of language comprehension, mathematical comprehension and mental effort are presented. These results are compared to those from program comprehension studies and similarities as well as differences are shown. In the next chapter, Chapter 4, we present the baseline tasks that are candidates to be used for *fMRI* studies of program comprehension. A selected subset of these baseline tasks is evaluated in an online study presented in Chapter 5 and evaluated in Chapter 6. Based on the results of the online study, we present an *fMRI* study design in Chapter 7. We then summarize the collected results in Chapter 8 and then discuss further work and studies that shall follow this work.



## BACKGROUND

---

In this chapter, we provide background information on terms that are important to understand research on and around program comprehension. These are in particular *functional Magnetic Resonance Imaging (fMRI)*, *Brodman Area (BA)*, cognitive load and program comprehension.

### 2.1 fMRI

In general, *functional Magnetic Resonance Imaging (fMRI)* is a non-invasive way to look inside a person's brain and observe the areas of the brain with higher blood flow. If a part of the brain has to work harder, for example during a task, it is supplied with more oxygenated blood. The different amounts of oxygenated and deoxygenated blood can be distinguished and therefore identified by the magnetic field. The deoxygenated blood contains deoxygenated hemoglobin. When exposed to a magnetic field, it can enhance the local magnetic field at that site, as opposed to oxygen-rich hemoglobin. This is called the *Blood Oxygenation Level Dependent (BOLD)* effect. This method is mainly used for the investigation of brain activity.

Regarding research purposes, the *fMRI* and *BOLD* procedure are a big step. Previously, participants had to reflect on their thinking and report what they were thinking during the study. This, of course, has an impact on the results of the study, because the participants were constantly interrupted in their thinking as they had to produce speech at the same time. With *fMRI*, this kind of think-aloud protocol is not needed and we can objectively measure cognition without further burdening the participant.

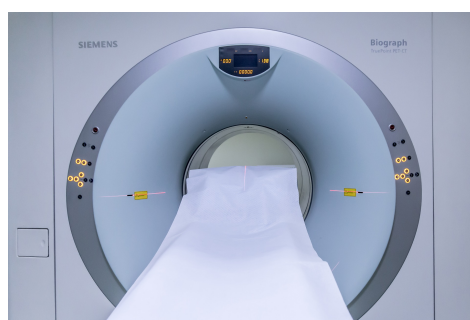


Figure 2.1: An *fMRI* device.<sup>1</sup>

However, around the *fMRI* scanner (Figure 2.1) it is important to note that the used materials must not contain any metal, as the scanner generates a very strong magnetic

---

<sup>1</sup> Picture from Michal Jarmoluk on Pixabay last visited: 26.03.2020

field. Any metallic object will be pulled against the magnet with incredible force and could injure the person and damage the scanner. The magnetic field has a strength of up to 3 Tesla in clinical applications and up to 10.5 Tesla for research. By comparison, the earth has a magnetic field of about  $10\mu$  Tesla [28]. Since the scanner is quite noisy and tight, a session should not last more than an hour [37]. Before the functional measurement of tasks, it takes the first 5 to 10 minutes to initialize the scanner because each brain has a slightly different size. Here the size and shape of the brain is measured so that later the functional activation can be mapped and identified. Afterwards the study can be started. However, rest breaks should be scheduled regularly during the study, as the work in the scanner is quite exhausting for the participant [36].

## 2.2 BRODMANN AREAS

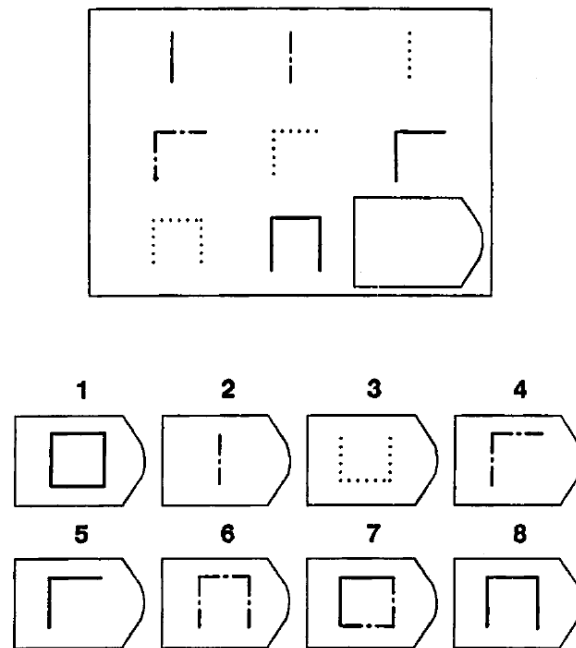
In 1909, the German neurologist, Korbinian Brodmann, examined the cells located in the brain. In the process, he sorted the cells according to size, density in relation to each other and the layer. Thereby he identified 52 different areas [16]. Brodmann's results are often used as a reference space in fMRI studies. The brain activations are projected onto the brain areas to analyze the functions and the interaction of areas. For example, it has been established that *Brodmann Area (BA)* 21 and 22, the *Wernicke area*, are responsible for word comprehension and word recognition [2]. Similarly, *Broca's area*, which is responsible for speech production, is already well localized. *Broca's area* is located in BA 44 and 45. In this thesis, the cognitive load and the program understanding will be examined in more detail. The general main goal of this research area is to be able to name specific areas for these tasks as well [46].

## 2.3 COGNITIVE LOAD

What is cognitive load and how can we make it accessible? In the paper by Baddley [3] working memory is explained as system with bounded capacity that enables temporary memorization and handling of information required for performing complex tasks like understanding, knowledge acquisition, and logical reflection.

In the paper by Whelan [44], the cognitive load is categorized into three categories. According to Sweller [43], these are *intrinsic load*, *extraneous load*, and *germane load*. The *intrinsic load* is generated when a concept or idea is understood. This concept or set of concepts should have a certain functionality that introduces complexity or *element interactivity*. By *element interactivity*, Sweller means that the individual elements interact so that the participant can build further understanding of the problem through their composition and interaction. As an example of *intrinsic load*, Whelan presents a paper by Newman et al. According to this paper, the riddle sentence "The first month after April is the month before my favorite month" generates a high intrinsic cognitive load. A more recent paper by Newman, Willoughby, and Pruce [26] on this topic is presented in more detail in Chapter 3.

However, it should be noted that in highly complex tasks, the distinctions between the three categories of cognitive load may be blurred.



**FIG. 1.** Illustrative Progressive Matrices item. Respondents are asked to identify the piece required to complete the design from the options below. (The item shown here is not from the current range of tests.)

Figure 2.2: An example of *RPM* as it is presented in the paper of Raven [33]

### Fascinated

- a) ill-treated
- b) poisoned
- c) frightened
- d) modelled
- e) charmed
- f) copied

Figure 2.3: An example of *MHV* as it was used in the TEDS studies, see [https://www.teds.ac.uk/datadictionary/studies/webtests/16yr\\_vocab\\_test.htm](https://www.teds.ac.uk/datadictionary/studies/webtests/16yr_vocab_test.htm)

According to Raven, there are two main components of cognitive load. He distinguishes between *educative* and *reproductive ability*. The *reproductive ability* describes the understanding and repetition of learned knowledge. The *educative ability*, goes in the direction of programming. In this case, the human being has to abstract concepts, understand the meaning of an initially confusing situation or use non-verbal schemes to solve complicated tasks. The *Raven Progressive Matrices (RPM)* test (Figure 2.2) was developed by Raven to measure the *educative ability*, the *Mill Hill Vocabulary (MHV)* test (Figure 2.3) is designed to evaluate the *reproductive ability*. The *RPM* is a non-verbal, well-validated test that does not use colors. This means that the test does not have to take into account red-green color blindness or any other weaknesses. In a 3x3 field 8 symbols are shown, whereby the field on the lower right remains empty. Then the participant has to recognize which symbol fits into the empty box. In the *MHV* test, 88 words are read aloud to the participant with increasing difficulty, and the participant has to explain or define them. The number of words varies depending on the age of the target group [33].

These papers provided a high-level overview of this area. Why we use *RPM* in the present study is discussed in Chapter 4.

## 2.4 PROGRAM COMPREHENSION

An example of high cognitive load is program comprehension, which is a problem-solving strategy [35], whereas a summary of cognitive processes is used. During program comprehension, the relevant parts of the source code must be found, therefore the source code must be screened and then the relevant information must be retrieved [39].

In the paper by Peitek [29], program comprehension is described as the underlying internal cognitive process that takes place when a programmer reads and understands source code. A distinction is made between the bottom-up and the top-down program comprehension model. Bottom-up is mostly used by novice programmers, where the program is analyzed line by line to gain an understanding of the higher-level functionality stepwise. This is a time-consuming and tedious process. However, if the programmer already has an idea of what the function does, based on his experience or the name of the function, the top-down model is used. At this point, the characteristic parts of the source code are examined to determine whether this functionality is actually implemented. Thereby beacons, (i.e., semantic cues [39]) such as well-chosen variable names, are of importance. Top-down comprehension is faster and exhibits lower cognitive load than understanding source code bottom-up.

The knowledge of program comprehension models helps software-engineering researchers to better understand the cognitive load of programming in the programmer's brain. This allows them to make adjustments to the *Integrated Development Environment (IDE)* that can reduce the programmer's cognitive load [35].

Further studies on this topic are presented in Chapter 3. In the study presented in this thesis, we have chosen bottom-up comprehension, which we explain in Chapter 4.

## RELATED WORK

---

After the background has been explained, we can now take a closer look at the research in this area. First, we show what is actually used as baselines in [fMRI](#) studies in neuroscience. Then we provide an overview of [fMRI](#) studies on mental load and language comprehension, which can inspire candidates for suitable baselines. Further, we present several studies in mathematics, as mathematical thinking is also believed to be a component of program comprehension. Finally, we discuss some program comprehension studies and their results.

### 3.1 BASELINES

Although [fMRI](#) is a major step ahead in neurobehavioral research, we cannot measure concrete brain activities of complex cognitive processes, such as program comprehension, in a high-quality way, because many underlying functions, such as reading itself, also cause activations in the brain. When the brain solves many subtasks at the same time, all of these areas are activated. In a complex task such as program comprehension, we cannot see exactly what belongs to the actual program comprehension and which areas have been activated for background tasks such as seeing or reading. Therefore, the question of a good baseline arises. Ideally, this would be the brain without any activities except those essential for survival. These breaks are typically necessary because the [BOLD](#) signal from the previous task has to drop before the next one is started. Therefore an experiment design often contains a rest condition. This rest condition is sometimes conducted with the eyes closed, sometimes with the eyes open and sometimes a cross is shown on which the participant should concentrate, also called *cross fixation* (Figure 3.1). However, the rest condition should not be used as a baseline. During this time the participants often exhibit reflective thinking, therefore sometimes even higher levels have been measured during the break than during the actual task [4].

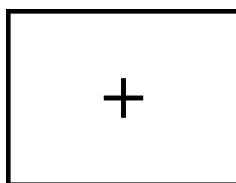


Figure 3.1: The *cross fixation* as it is used in [fMRI](#) studies

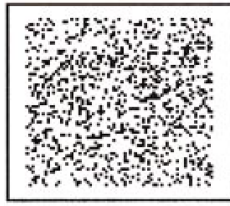


Figure 3.2: The *visual noise* used in fMRI studies by Martin [22]

In the paper by Kroger et al. [18], many studies on deductive processing are compared. It includes a table in which the tasks, the control task or baseline condition, and the outcome are shown. In this table it becomes clear that the outcome is always different depending on the specific task as well as the baseline used. This further highlights the importance of finding suitable baselines for fMRI studies in general.

The paper by Martin [22] examines the *Medial Temporal Lobe (MTL)* areas. The actual study is less relevant for our research than the finding that the rest condition cannot be used as a baseline. The MTL areas were also activated in the rest time. Therefore, if activity in the MTL areas is to be examined, the baseline must be different from rest. This study therefore used *visual noise* (Figure 3.2) as it distracts from the current situation, namely lying in an fMRI scanner [23].

In the paper by Stark, and Squire [42] several small tasks were selected, for example, odd/even and left/right judgments for numbers and arrows. They also found stronger activation in some brain areas during the rest condition than for example, during the odd/even task. Thus, the rest condition is not a good baseline as daydreaming or self-reflection kicks in. This issue is also relevant for program-comprehension studies, as also shortly mentioned in the paper by Peitek et al. [30].

Similarly, in the study by Culham, Cavanagh, and Kanwisher [8], it is mentioned that non-controllable cognitive behavior could arise when simply passively viewing the baseline condition without interaction. Thus, our baseline candidates must include some interaction to actively engage participants.

In the paper by Binder et al. [4] different tasks and their preliminary stages are compared against rest. First, the participants rest with their eyes closed. Then, they listen to four sounds and click, if there are at least two high-pitched sounds in this sequence. Then a semantic task in which animal names were read aloud and the participant should respond if it is a U.S. farm animal. Lastly, a phonetic task where the participant was asked to respond when both b and d were pronounced in a meaningless three-syllable word. Again, the rest was considered as an inappropriate baseline, since "thinking about problem solving regularly occurs during resting states" [4] and thus activated areas were not found, since they were also activated during the baseline. It was interesting to note that *task-unrelated thoughts (TUT)* were documented. The subjects were asked to state honestly, if they had thought about anything else during the rest. This was the case for all 14 participants, although they were made aware of the existence of such in advance. Consequently, this paper concludes that

"the conscious resting state is not a state of neural inactivity" [4]. Therefore, it is suggested to use similar tasks that lack exactly the detail to be investigated as baselines. In this work by Binder, this would be the phonetic task compared to the semantic task for finding the areas involved in semantic processing. In this thesis, we aim to find a similarly suitable baseline.

Since countless baseline possibilities have been used in research, it is clear that there is no universal baseline for every study. Rather, one should select a baseline specifically on the task under investigation. Thus, we compile baseline candidates for program comprehension from reading, arithmetic and mental load. Prior to testing this in time-consuming and expensive studies, we conducted a literature review of the activated [BAs](#) to determine whether the outcome of the study could be expected to be beneficial.

The next sections present further related work for the areas that are baseline candidates. For this purpose, [fMRI](#) studies on mental load, language, mainly reading, and mathematics, especially arithmetic, as well as program comprehension were analyzed. For each of these studies we ask the following questions:

- What task type does this study use?
- If specified, what task type does it use as a baseline?
- Which [BAs](#) are activated for the main task in this study?
- Is there anything else we can learn about baseline issues from this study?

The decision to take a closer look at these domains is because they are believed to form the cognitive basis of programming. In textual programming languages, the human brain must read the text as a prerequisite for understanding the underlying program operations. In many programs, mathematical operations are performed, so we are also interested which areas of the brain perform mathematical tasks. In addition, programming is a complex cognitive activity, which generates a relatively high mental load.

We present the studies performed in these areas, with the activated [BAs](#), to show that the areas overlap. This motivates us to take reading, mathematics and mental load into the baseline for program comprehension so that we can explore the differences in brain activity in further studies. To obtain an idea how this can be used, future studies with our baseline could then investigate, for example, the influence of variable names or syntax highlighting or anything else that could influence the cognitive load during program comprehension.

## 3.2 FMRI STUDIES ON LANGUAGE PROCESSING

In this section, we take a look at selected [fMRI](#) studies on language processing (Table 3.1).

Paper	Task	Baseline	Identified <a href="#">BA</a> s
Newman [25]	incorrect sentences (syntax)	correct sentences	6, 8, 21, 22, 41, 43
Newman [25]	incorrect sentences (semantic)	correct sentences	6, 8, 9, 10, 21, 24, 31, 39, 46
Krueger [20]	writing prose	writing source code	17, 18, 21, 22, 44
Bonhage [5]	normal sentences	scrambled sentences	4, 6 – 9, 18 – 24, 31, 32, 37, 40, 43
Price [32]	Review: language processing		44, 45, 47
Hagoort [12, 13]	Review: language processing		6, 44, 45, 47

Table 3.1: An overview of the presented studies and review papers on language processing

There is already an abundance of results in the field of language processing, therefore two literature review papers representing the relevant studies mentioned in them are presented. In the paper by Price [32] a large number of studies are mentioned and compared. The prominent areas in language processing are [BA](#)s 44, 45, also known as *Broca's area*, and also [BA](#) 47. Furthermore, the papers by Hagoort [12], and Hagoort and Indefrey [13] similarly summarize the results of many studies. According to them, language production and processing takes place in [BA](#)s 6, 44, 45 and 47.

In the study of Newman et al. [25] syntactically or semantically incorrect sentences (Listing 3.1) were presented to the participants. This type of language testing is very popular in experiments using *Electroencephalography* ([EEG](#)) to measure *N400*, a negative peak of the measurements after 400 *milliseconds*, and *P600*, a positive peak of the measurements after 600 *milliseconds*. The brain has to work harder in the language areas to compensate the ambiguities or errors in the sentences. In this study, [fMRI](#) data has been collected. The syntactically or semantically incorrect sentences were evaluated against a baseline of syntactically and semantically correct sentences. Based on this, [BA](#)s 6, 8, 21, 22, 41, 43 are involved in syntax and [BA](#)s 6, 8, 9, 10, 21, 24, 31, 39, 46 in semantic processing.

Listing 3.1: Two example sentences from the paper by Newman et al. [25]

correct:	Yesterday I cut Max's apple with caution.
Syntax incorrect:	Yesterday I cut Max's with apple caution.
correct:	Yesterday I sailed Todd's boat to China.
Semantic incorrect:	Yesterday I sailed Todd's hotel to China.

Furthermore, in the study of Krueger et al. [20] on code writing, the areas activated in the long prose response were also identified. The participants wrote a prose text in the [fMRI](#) device using a special keyboard. The activated areas are [BA](#)s 17, 18, 21, 22, 44 given the



baseline of writing source code. We will look at this paper in more detail when presenting papers on program comprehension.

Additionally, the study by Bonhage et al. [5] monitored eye tracking and fMRI simultaneously. They observed the presence of predictions in reading by showing normal and nonsense sentences to the participants. The nonsense sentences had a sentence-like structure, as they were normal sentences in which the letters of the words were scrambled. Thus, a general syntactic structure was preserved. The brain activation measured in the normal sentences while using the nonsense sentences as baseline are in the BAs 4, 6 – 9, 18 – 24, 31, 32, 37, 40, 43.

Depending on the baseline used, different areas were found. In the literature reviews by Price [32], Hagoort [12], and Hagoort and Indefrey [13], the areas that were searched were mainly those which were involved in reading. In the other papers presented [5, 20, 25], the focus was on more advanced questions, such as prediction, filling in gaps or finding syntax errors as well as semantic errors. Accordingly, normal reading was chosen as the baseline. Consequently, the expected reading areas do not appear in their results.

### 3.3 FMRI STUDIES ON MATHEMATICS

Next, we address the mathematical basics by reviewing fMRI studies on mathematics (Table 3.2).

Paper	Task	Baseline	Identified BAs
Newman [26]	math equations	cross fixation	6, 7, 40
Newman [26]	math equations	word riddles	13, 32, 40
Zago [45]	calculate	read numbers	6, 7, 9, 18, 37, 39, 40
Krueger [19]	integral verification	font verification	-

Table 3.2: An overview of the presented studies on mathematics

The study by Newman, Willoughby, and Pruce [26] investigated whether mathematical problems are easier to process in text form or in the form of equations. This study explored the difference between word and number riddles. We will also present this study in the section on mental load, Section 3.4, but we present it here too, since it also relates to mathematical thinking. This underlines why it is important to obtain an overview of the different areas, since the boundaries between them are fluid.

In the study it was noticed that although the tasks were different, similar brain areas were activated, which resulted in an overlap of the activated areas. The word tasks were word riddles and the mathematical tasks were the equational equivalent (Table 3.3). Easy and difficult tasks were distinguished to capture the influence of difficulty of the tasks. Comparing mathematical tasks, both the easy and the difficult ones, there was no significant difference. When evaluating against *cross fixation* the BAs 6, 7, 40 were activated. In comparison the mathematics tasks evaluated against the baseline, which was chosen as the word riddles,

Level	Word riddle	Equation
easy	The month after April is the month before my favorite month.	$1 + 4 = x - 1$
hard	The month before my favorite month is the month after April.	$x - 1 = 4 + 1$

Table 3.3: An example of the word riddles and the corresponding equations, as it is presented in the paper by Newman, Willoughby, and Pruce [26].

activated the [BAs](#) 13, 32, 40.

Without judging the appropriateness of these baseline choices, it shows once again how important the information, which baseline was used, is in studies, since the results presented lie in clearly different areas. The used baseline should always be clearly recognizable.

In the study of Zago et al. [45] the participants were asked to calculate mathematical problems. The reading of numbers was taken as a baseline. Simple mathematical problems were calculated, for example, single-digit numbers were multiplied, as well as more difficult tasks, in which two-digit numbers had to be multiplied whose result was less than 1000. Thereby, a difference between the simple and difficult mathematical tasks was recognized, since the simple mathematical tasks were not calculated but reproduced. This means that simple math problems are answered with the help of the memory, maybe they were learned by heart at some point in time, like the small multiplication table. In the difficult tasks, the participants had to use solution strategies and the intermediate results had to be stored in the brain and recalled. Contrary to previous studies, no language areas such as the *Broca area* or the *Wernicke area* were activated. This might also be related to the baseline used. The activated areas in the difficult tasks with baseline "read numbers" were converted to [BAs](#) to compare them with the other studies using the *Montreal Neurological Institute (MNI)* coordinates. The areas were converted using the *Yale MNI to Talairach with Brodmann Areas* website<sup>2</sup>. The [BAs](#) activated in the presented study are 6, 7, 9, 18, 37, 39, 40.

In the paper by Krueger et al. [19] the participants were asked to calculate an integral and to verify, if the solution displayed during the calculation is correct. As a baseline task, they were asked to verify whether the same font was used in the integral and in the possible solution. The brain areas were evaluated against the described baseline. It was recommended that the next time they should additionally solve simple mathematical tasks to receive a better understanding of which areas are used for the more appealing mathematics. This paper deals with the choice of a good baseline, but unfortunately the areas are not provided in [BAs](#) and therefore we cannot compare them with the other studies.

This has provided us with an overview of the mathematical studies. We should be careful not to choose tasks that are too easy, because they are reproduced and not calculated [45]. With respect to baselines, we have again found similar tasks, to the actually investigated tasks. In each case, the visual and motor parts are very similar, only the difficulty changes.

<sup>2</sup> <http://sprout022.sprout.yale.edu/mni2tal/mni2tal.html>

## 3.4 FMRI STUDIES ON MENTAL LOAD

We start with presenting a selection of papers related to mental load (Table 3.4).

Paper	Task	Baseline	Identified BAs
Prabhakaran [31]	analytic RPM	match RPM	6, 7, 9, 10, 18, 19, 21, 32, 37, 39, 40, 45, 46
Prabhakaran [31]	figural RPM	match RPM	7, 9, 19, 32, 37, 40, 46
Prabhakaran [31]	analytic RPM	figural RPM	6, 7, 9, 18, 19, 21, 37, 39, 40, 44, 45, 46
Newman [26]	difficult riddles	easy riddles	6 – 10, 40
Cohen [7]	rotated cube figures	rest	1 – 4, 6, 17, 18, 19
Cohen [7]	rotated cube figures	mirrored cube figures	1, 2, 3, 7, 8, 19, 39, 44, 46
Culham [8]	observe moving dots	moving dots	-
Baddley [3]	Review: visuospatial sketchpad		6, 19, 40, 47
Baddley [3]	Review: phonological loop		40, 44

Table 3.4: An overview of the presented studies and review papers on mental load

In the review of Baddley [3] the working memory is described as, temporal memory with the ability to process the data. Working memory allows more complex tasks to become possible. The central processing is supported by two components, on the one hand the *visuospatial sketchpad* BAs 6, 19, 40, 47 and on the other hand the *phonological loop* BAs 40, 44.

Otherwise, in the study of Culham, Cavanagh, and Kanwisher [8] participants were asked to observe moving dots. While doing so, they were asked to fixate on the center of the screen. The dots moved randomly and only those that were initially orange should be observed. In the baseline, the green dots flying around without any dots to observe were used to see the cognitive load enhanced. As the number of observable dots increased, the cognitive load increased. Unfortunately, the data on brain activation were not provided with BAs, so we can only use the study design here as an example study to compare the procedure and not the activated areas.

In the study by Prabhakaran et al. [31], three categories of the RPM were shown to participants. As a baseline, a 9x9 field, which was randomly created without rules, was used that already had an answer on the bottom right. Then, the participant only had to click on the same figure as on the bottom right at the answer panel. This baseline task is called *match* (Figure 3.3). The other two tasks included rule-based transformation within the tiles (e.g., a flower gets more and more leaves), namely the *figural* task (Figure 3.4). And analytic problems, where the participant has to recognize regularities in patterns, called the *analytic* task (Figure 3.5). The tasks should represent *analytic reasoning*, *figural* or *visuospatial reasoning* and *simple pattern matching*. Thereby *fluid reasoning* which is the capacity to identify and

perceive relational links between any foundations should be investigated. The BAs which were found in the process are all related to working memory. In the evaluation, *analytic* vs. *match* activated the BAs 6, 7, 9, 10, 18, 19, 21, 32, 37, 39, 40, 45, 46. Whereby *figural* vs. *match* showed activation in the BAs 7, 9, 19, 32, 37, 40, 46. The analysis of *analytic* vs. *figural* revealed higher activation in BAs 6, 7, 9, 18, 19, 21, 37, 39, 40, 44, 45, 46 for the *analytic* condition.

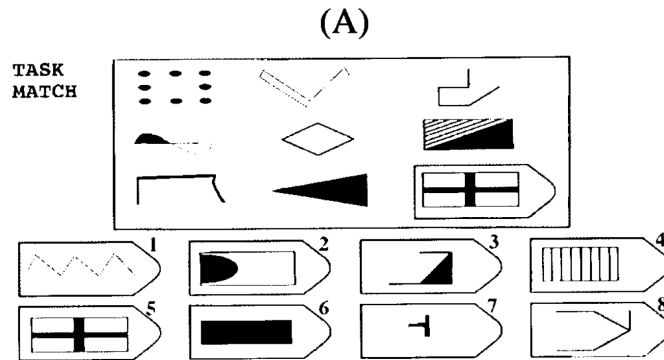


Figure 3.3: The *match* task from the paper of Prabhakaran et al. [31]. This matrix does not follow any rules and the answer is already filled in the 9x9 field. The participant only needs to select the answer no. 5 in this example.

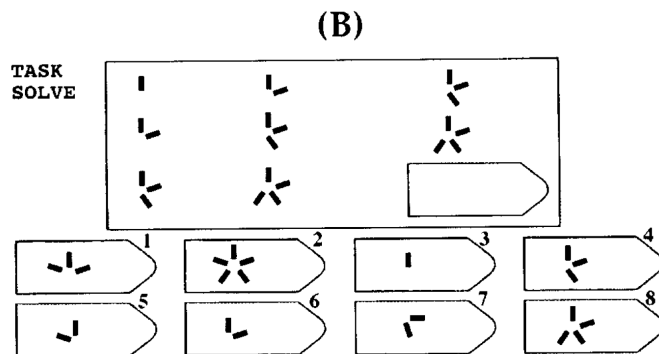


Figure 3.4: The *figural* task from the paper of Prabhakaran et al. [31]. In this example the number of leaves of the flower increases right to left and top to bottom. These rules result in the correct answer no.2.

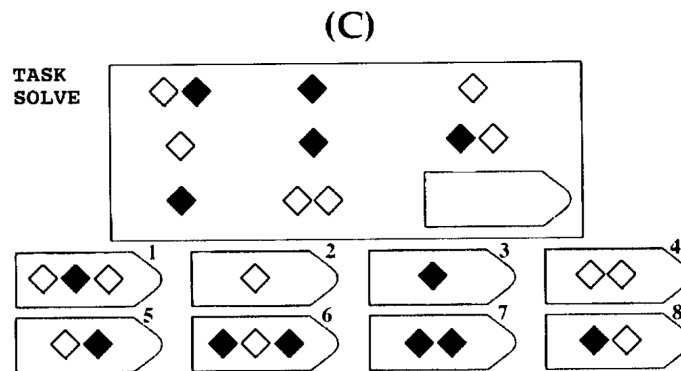


Figure 3.5: The *analytic task* from the paper of Prabhakaran et al. [31]. In these tasks different types of rules were applied, for example subtraction, addition, distribution of 2, and distribution of 3.

The study by Newman, Willoughby, and Pruce [26] explored the difference between word and number riddles (Table 3.3). There were multiple components tested against each other, whereas we are interested in *hard > easy*. Thus, the easy tasks are the baseline and the hard tasks are the actual task. The activated brain areas are BAs 6,7,8,9,10,40. However, differences in difficulty were found especially in the word riddles. For the number riddles, no significant difference in difficulty was found. Remarkably, BA 17 is activated against the rest. This area belongs to the visual cortex [17]. In the comparative analyses against other tasks, this area was not identified.

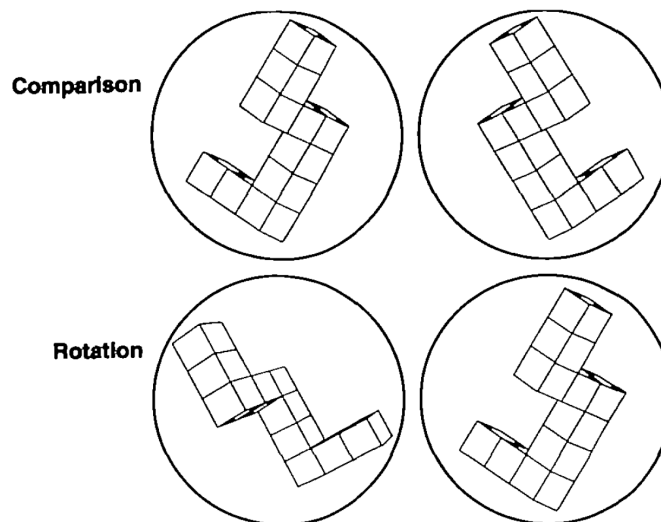


Figure 3.6: The two different cube figure tasks from the paper by Cohen et al. [7]. The comparison task asks if the same cube figure is shown on the right side, not the mirrored version. The second task is the rotation, here the participant has to decide if this is the same cube figure but shown from different angles.

Additionally, in the study by Cohen et al. [7] participants were asked to compare two pictures of figures made out of cubes (Figure 3.6). Participants had to decide whether they

were the same figure that had only been rotated or whether they were different figures. The baseline condition was also a task in which two pictures were shown. Here, either the identical cube figure was shown twice or exactly mirrored. In this task, the participants were asked to decide whether it was identical or not. Overall, the study was evaluated against the rest condition, but we are mainly interested in the evaluation against the baseline. The [BA](#)s 1, 2, 3, 4, 6, 17, 18, 19 were detected, when evaluating against rest. During rotation, in the evaluation against the baseline, the participants had stronger brain activity, although these were not stronger than the activation measured against the rest condition. In the evaluation against the baseline, the [BA](#)s 1, 2, 3, 7, 8, 19, 39, 44, 46 were activated.

To summarize, in the presented studies on mental load, the baseline was a task that had a very similar visual structure [[7](#), [8](#), [26](#), [31](#)] to the actual task, and included the same motor demands as the actual task [[7](#), [31](#)].

## 3.5 FMRI STUDIES ON PROGRAM COMPREHENSION

In the following, we provide an overview of the [fMRI](#) studies on program comprehension (Table 3.5).

Paper	Task	Baseline	Identified <a href="#">BAs</a>
Siegmund [38]	<i>Java</i> comprehension	syntax error search	6, 21, 40, 44, 47
Peitek [30]	<i>Java</i> comprehension	rest	6, 21, 40, 44, 47
Krueger [20]	fill-in-the-blank code	fill-in-the-blank prose	1 – 4, 6 – 9, 13, 20, 21, 22, 37
Krueger [20]	long code writing	long prose writing	7, 19, 39
Krueger [20]	code writing (both)	prose writing (both)	4 – 6, 9, 10, 13, 18, 19, 24, 32, 39, 40
Castelhamo [6]	C comprehension	pseudo code	6, 9, 19, 21
Huang [14]	data structures	mental rotation	8, 21, 22, 31, 32, 38, 39
Liu [21]	<i>Python</i> comprehension	"fake code"	6, 7, 8, 9, 10, 22, 37, 44, 46, 47
Siegmund [39]	bottom-up (no beacons)	syntax error search	6, 21, 40, 44
Siegmund [39]	top-down (beacons)	bottom-up (no beacons)	39
Siegmund [39]	bottom-up (no beacons)	top-down (beacons)	6, 21, 40, 44
Floyd [10]	C comprehension	plain English text	-

Table 3.5: An overview of the presented studies on program comprehension

We start with the first [fMRI](#) study on program comprehension, namely the paper by Siegmund et al. [38]. In this study, *Java* code snippets were shown and the participants were asked to identify its functionality. The source codes snippets did not contain any comments or helpful variable names, so that a bottom-up understanding takes place. Similar source codes snippets were used as a baseline. Now the task was to find syntax errors, such as missing parentheses. The functionality of the source code did not matter in this case. The activated areas of the brain were given in [BAs](#) 6, 21, 40, 44, 47. In the paper it is concluded that the areas used in program comprehension are related to working memory and language.

In the study by Peitek et al. [30], participants had to generate the output of *Java* programs and search for syntax errors in *Java* programs. The researchers concluded that a *Java* experienced participant has less cognitive load during program comprehension than a novice. As a baseline they use rest condition and as a contrast task they used the syntax error search. In their evaluation of the [BOLD](#) Signal, they overlapped the program comprehension line and the syntax error line, to show similarities and differences. They found the [BAs](#) 6, 21, 40, 44, 47 activated during program comprehension. For all areas the contrast task, syntax error, has a

lower **BOLD** signal, except for BA 47 where both **BOLD** signals are almost identical. According to their research, the **BA**s 6, 21, 40, 44, 47 related to program comprehension are associated with working memory, attention and language processing.

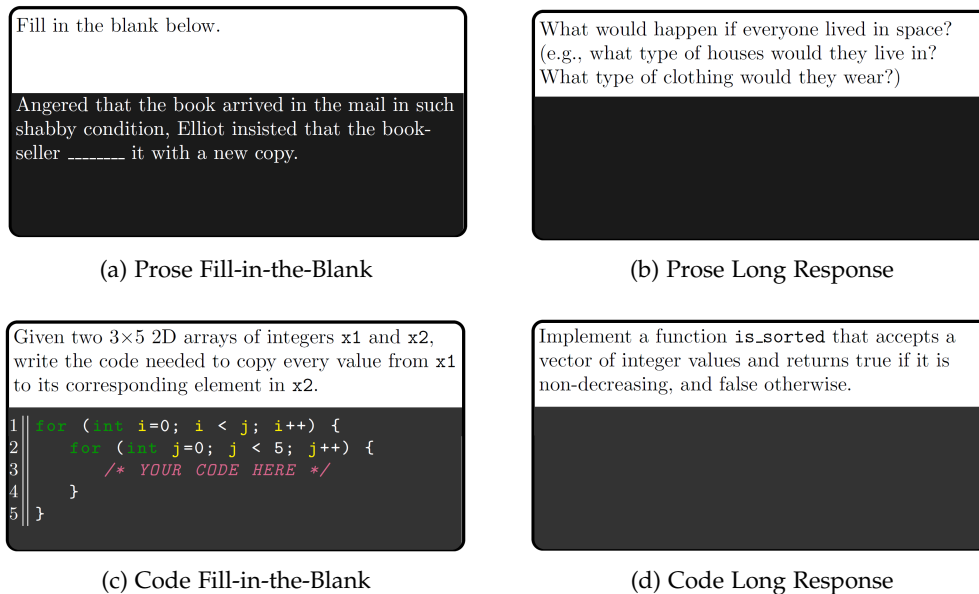


Figure 3.7: The tasks of the study by Krueger et al. [20], as they are presented in the original paper. In the Prose condition the participant should write normal English words or sentences. In the code condition the participant is asked to write *Java* code. The condition *Fill-in-the-Blank* means, that the solution is already given except for one word or one line of code. In the *Long Response* it's all up to the participant to solve the question, without a partly given solution.

In the paper by Krueger et al. [20] several contrasts are examined in one study. The participants should write both prose text and source code. In detail, they were shown *Fill-in-the-Blank* texts that they were asked to complete or were expected to give a longer answer. These combinations make four different types of tasks (Figure 3.7). In the evaluation three different combinations of *task* and *task used as baseline* were chosen. First, all code writing tasks, whether *Fill-in-the-Blank* or full source code, were compared against all prose writing tasks, again both *Fill-in-the-Blank* and longer text answer. This results in the activated **BA**s 4 – 6, 9, 10, 13, 18, 19, 24, 32, 39, 40. Then the *Fill-in-the-Blank* tasks on source code and *Fill-in-the-Blank* with prose were compared. Hereby activation in the **BA**s 1 – 4, 6 – 9, 13, 20 – 22, 37 was found. Finally, the longer answer written in prose and the one written in code were evaluated against each other. There the **BA**s 7, 19, 39 were activated. In this study, an **fMRI** appropriate keyboard was used. It is important to note here that this is about the writing, most other studies have mostly investigated the reading of code due to technical limitations in an **fMRI** scanner.

In the study of Castelhana et al. [6], participants were presented with C programs containing bugs and were asked to find the bugs. The main focus of this study is the brain activation during bug detection and verification, but also the activation during program



comprehension was assessed. For this purpose, the activities during pseudo code reading and C code reading were considered. In the evaluation they compared source-code against pseudo code, whereby the baseline is the pseudocode. The activated **BA**s are 6, 9, 19, 21.

In the paper by Huang et al. [14], mental rotation tasks were examined with respect to their different brain activity in data structure tasks (e.g., lists, arrays, and trees, see Figure 3.8). In general, it is found that programming uses similar areas but with stronger activation, and also the cognitive load increases faster. In this study, not only **fMRI** but also *functional Near-Infrared Spectroscopy* (**fNIRS**) was used, whereas **fMRI** turned out to be more useful, because more brain areas can be captured. Thus, more areas that were activated could be found. The activated brain areas are **BA**s 8, 21, 22, 31, 32, 38, 39.

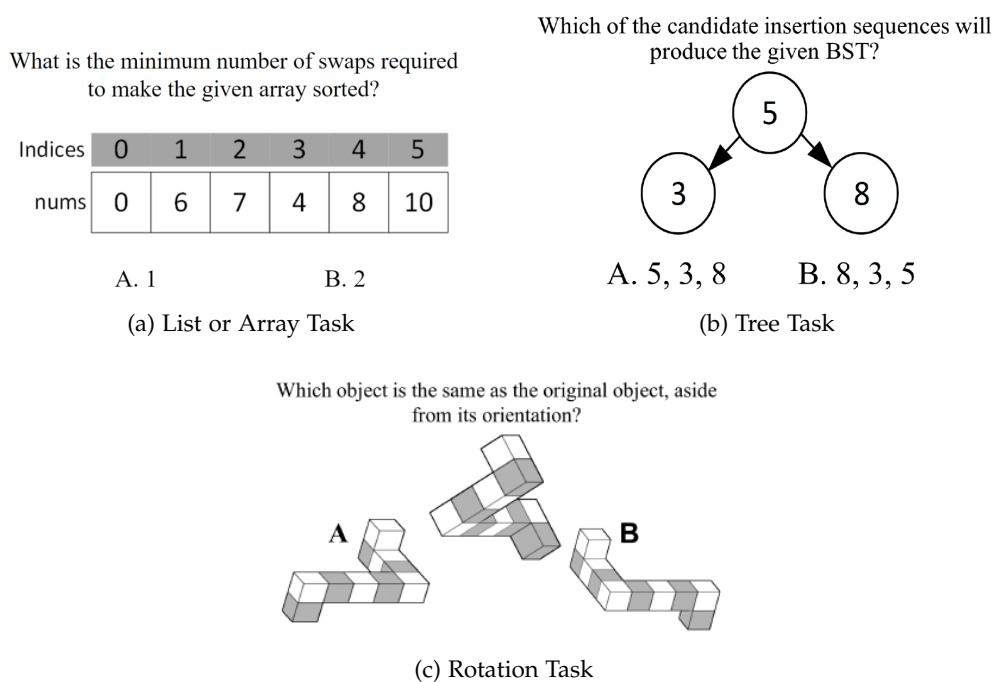


Figure 3.8: The tasks of the study by Huang et al. [14], as they are presented in the original paper.

In the paper by Siegmund et al. [39], the influence of beacons was investigated in more detail. For this purpose, the code snippets were presented to the participants in advance so that they would not do bottom-up comprehension in the scanner, but top-down comprehension using beacons. In the scanner itself, the code snippets were presented in four different ways, with and without beacons and with and without proper layout. The baseline used here was the detection of syntax errors, as described in the previous paper. Now, the bottom-up comprehension, without beacons, was evaluated against the syntax error detection. The **BA**s 6, 21, 40, 44 were activated. Furthermore, bottom-up comprehension without beacons was compared to top-down comprehension with beacons, where **BA**s 6, 21, 40, 44 were activated for bottom-up comprehension. Here **BA** 39 was deactivated, this means that **BA** 39 is activated at top-down comprehension.

In the study of Liu et al. [21] language, mathematics, logic, and localizer tasks of the multi-demand systems were performed and compared with the areas activated during program comprehension. In the evaluation on code comprehension in *Python*, whereby "fake code" was used as baseline, the *BA*s 6, 7, 8, 9, 10, 22, 37, 44, 46, 47 were activated. A "fake code" consisted of a normal function, whereby it was completely scrambled (Figure 3.9). The results of the other tasks were unfortunately not given in such detail that they can be interpreted as *BA*s. However, it is explained that the areas of program comprehension significantly overlap all other tasks, mathematics, language, logic and localizer task. The overlap is most significant for logic with baseline language.

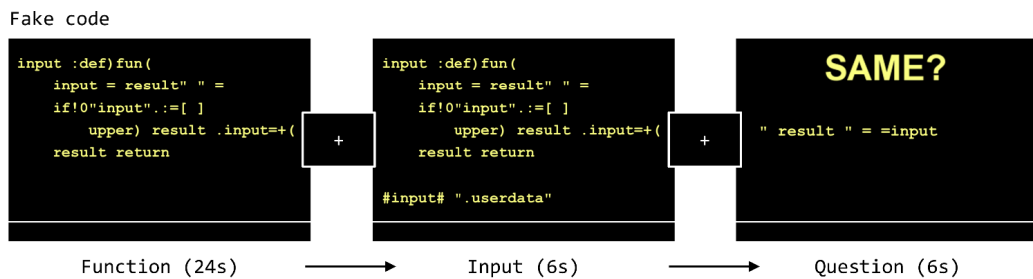


Figure 3.9: The "fake code" condition from the paper of Liu et al. [21]. This is only an example and does not correspond to the actual vision of the participant in the *fMRI* scanner.

Not every research on program understanding deals with the localization of brain areas. We are thinking outside the box and showing here two other possibilities based on papers of what else can be measured. These studies show different but also similar research directions. Of course, such studies can also benefit from finding a generally used baseline.

In the paper by Ikutani et al. [15], *Java* code snippets were shown and the implemented algorithms should be classified into 4 different groups. The participants were selected because they had performed well in programming competitions. The competition performances were correlated with the *fMRI* data from the scanner. The results showed that, in the competition, good programmers process code snippets and prose more similarly in the brain than weaker programmers.

In the paper by Floyd, Santander, and Weimer [10], the processing of plain English text was compared with program comprehension. For this, participants were asked to make statements about whether the proposed output matched the *C* snippet and read and accept or reject code and prose review comments. This showed that the language region operates differently in the code vs prose tasks. A classifier was trained to guess the different tasks based on brain activity. It is noticeable that the same areas are considered to distinguish programming and prose. The difference between them becomes smaller the more experienced the programmer is.

### 3.6 SUMMARY OF THE RELATED WORK

Across all presented papers, it is noticeable that the results are different depending on the baseline used [18]. The baselines in language papers were sometimes also reading, without syntax errors or ambiguities. In these papers, where the baseline was also reading, the BA 44 and 45 are not listed as results. These papers examined linguistic anomalies or similar issues, so by omitting reading for simple sentences, they were able to better capture brain activation in ambiguities [25]. Likewise, with the mental load tasks, the papers of [7, 26] also activate area 17 during *cross fixation* or rest baseline. This belongs to the visual cortex and is responsible for the visual input [17].

In mathematical papers, we do not observe such a trend in our selection. In program comprehension papers, we note that in Krueger et al. [20] none of the language areas were activated because the baseline is natural language text. Liu et al. [21] found activation in these areas, however, because possibly the fake program code was noticed without real reading and the participants therefore did not really reread it, but already classified it by the appearance by strange brackets etc. as a fake task or did not have so much mental concentration for the more exact reading. In the experiment by Lui, the logic tasks took the longest, followed by the mathematical tasks and the language tasks went the fastest. Nevertheless, Liu was able to show that the areas, which were activated, overlapped with the areas of the localizer tasks (mathematics, language, logic) more than just by chance. The most overlapping areas were logic, followed by mathematics and finally language.

It has now become clear that the baseline should be chosen to cover everything that is not part of the brain effort that we want to investigate. For example, if the study is related to the comprehension of language, the baselines for language are often the visual perception of writing and the reading of individual letters or words. In mathematics, the baselines were often a recognition of numbers, for example, deciding whether there is a one in a series of numbers [34]. For mental load, Prabhakaran et al. [31] used a "click the same symbol"-task as a baseline that had the identical layout to the other tasks. In all studies, the motor activity that occurs when the button is pressed is also included in the baseline.

Thus, a baseline in these areas is a task where the layout of the task is shown visually, with letters, numbers or pictures, depending on what is used, and that a button is also clicked as in the real tasks.

Therefore, the question arises: What is program comprehension? Seeing the task is definitely part of the baseline. As well as the reading and semantic understanding of the variable names. Additionally, the calculation of indices like  $i = j + 5$  is not necessarily sufficient to understand the overall functionality of the program. Also, the recognition of patterns, like the indentation of *if-statements* does not belong to the actual understanding. It is possible to imagine a human being who can read the variable names like *sum* and *result* and recalculate the calculations like  $i = j + 1$  and  $result = sum/2$  and also recognize the *while loop* and register that this part is repeated. However, this human may not be able to abstract and conclude that this program adds the numbers from 0 to the given input.

This abstraction, especially with recursive calls, goes beyond reading, simple arithmetic and recognition of patterns. Either the same areas are more activated, or other areas are working on these problems, or both. This can be investigated by choosing the baseline tasks in such a way that only the step of program understanding remains.

Consequently, we want to propose tasks from the areas of arithmetic mathematics, reading, and mental load as baseline tasks.

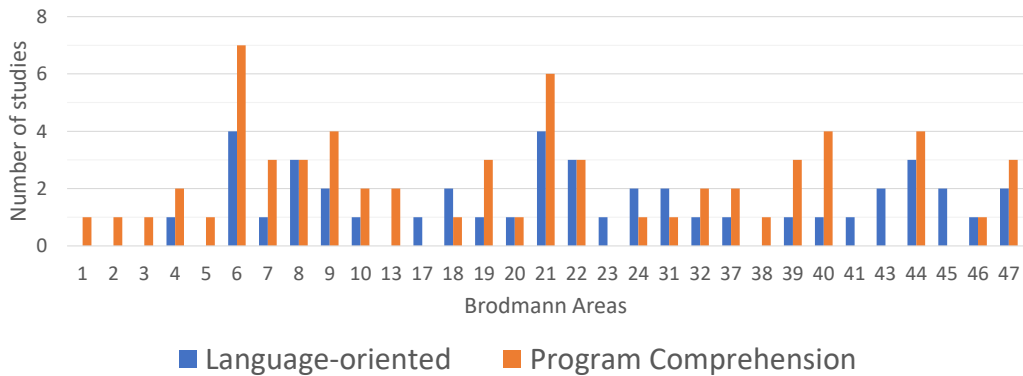


Figure 3.10: The activated areas in language-oriented studies are shown in blue and the areas activated during program comprehension are colored in orange. The areas that were not activated in any of the studies considered have been excluded (e.g., number 11).

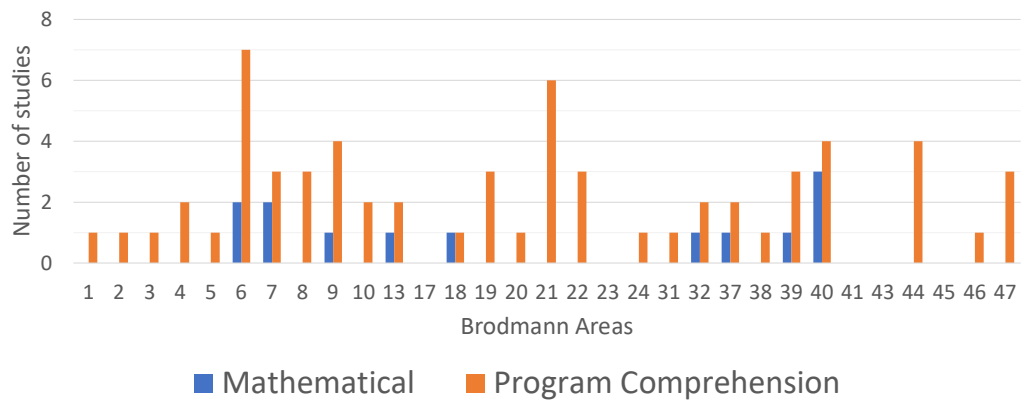


Figure 3.11: The activated areas in math-oriented studies and during program comprehension

To provide a better overview of the activated areas, we present them as graphs (Figures 3.10, 3.11 and 3.12) showing the number of activated areas in the presented areas. For a frequency analysis we have looked at too few studies, but the diagrams help to get an overview. The BA 6, 21, 44, 47 of the program comprehension could be covered with the language-oriented tasks. The BA 6, 40 are very similar to mathematics. Furthermore, BA 6, 7, 9, 18, 19, 32, 37, 40, 44 overlap with mental load. Thus, we can assume that an fMRI study with these tasks can provide helpful results.

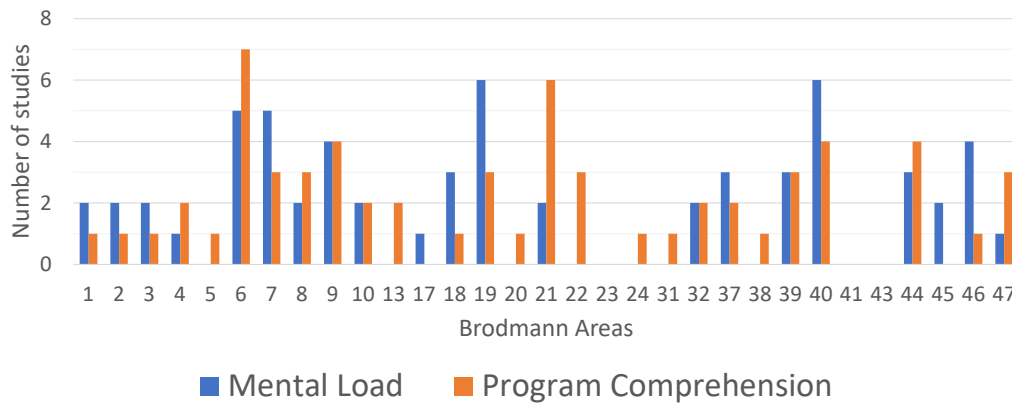


Figure 3.12: The activated areas in mental-load studies and during program comprehension



## SUGGESTED TASKS

---

In the previous chapter of the related work, we presented many papers from the areas of the *fMRI* research. We summarized the results of the studies in an overview. Then, based on that, we developed the tasks for possible baseline conditions in program comprehension. Now we present the exact tasks that were used in our study.

First of all, the tasks for the various areas should have approximately the same difficulty as the program tasks, which will be ensured by evaluating the processing time from the prestudy. They should therefore be comparable in difficulty. This excludes small tasks such as "is there a 1 in the program or not", since they do not evoke any deeper thoughts. In the following, the possible baseline tasks for the respective categories are presented.

### 4.1 LANGUAGE-ORIENTED TASKS

#### Listing 4.1: Example of a language-oriented task

The searched term refers to the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. Since the mid-20th century, scientists have attempted to develop a system capable of carrying out tasks perceived as requiring human intelligence. Among the tasks that have been studied from this point of view are game playing, natural-language understanding, fault diagnosis, robotics, and supplying expert advice.

- a. artificial intelligence
- b. computer
- c. robot
- d. self-driving car

On the *Britannica Kids* website<sup>3</sup>, there are short English texts about objects or things. In these texts we have used the preview and slightly rephrased it, so that the word the text is about does not appear in it. For example, *the searched term* or *it* instead of the word itself was used. Additionally, we shortened some sentences or added new ones to create texts of approximately the same length without compromising their comprehensibility. The modified texts are about the same length namely between 65 and 82 words. We also aimed at avoiding ambiguous sentences. The 4 answer choices consist of the correct word plus 3 other meaningfully similar words (Listing 4.1). All used tasks can be found in the Appendix A.

---

<sup>3</sup> <https://kids.britannica.com/students/browse/subjects>, last visited: 15.12.2020

## 4.2 MATHEMATICAL TASKS

Listing 4.2: Example of a mathematical task generated by our python script

$$8 - 7 - 9 - 6 + 2 + 3 - 2 = ?$$

- a. -11
- b. 28
- c. -26
- d. -6

The mathematical task is an arithmetic task in which seven single-digit numbers must be added and subtracted (Listing 4.2). It is important to note that the calculation with seven numbers is not automated cognition, such as the calculation of two single-digit numbers. Davis et al. [9] revealed that adults do not calculate simple arithmetic tasks, but reproduce them by recollection. Therefore, we emphasize not too simple arithmetic tasks. We use only addition and subtraction, because with the extension of multiplication the rules of arithmetic would have to be considered. These are known to everyone, but since they are not consistently implemented everywhere (for example, the windows calculator <sup>4</sup> or many cell phone calculators) this could lead to complications.

During the mathematical task, the participants first see the arithmetic calculation, then they are shown 4 possible answers, one of which is correct. These tasks were randomly generated using a *Python* script (Listing 4.2). All used tasks and the *Python* script are given in the Appendix A.

## 4.3 MENTAL LOAD-ORIENTED TASKS

In the Background, see Chapter 2, two theories of mental load were presented, each with an example study for the part of the cognitive load that includes program comprehension. For this work, we have to decide on one approach. We choose the *Raven Progressive Matrices (RPM)* because it is a non-verbal task. With this work, we aim at activating three areas in the brain: mathematics, text comprehension and mental load. This enables us to compare them against the brain activation of program comprehension. Thus, these three areas must be as clearly differentiated as possible. Therefore, we decided against text puzzles and for *RPM*.

The task of an *RPM* is to find the 9th symbol in a 3x3 field with 8 symbols, according to the rules used. The symbols that are available for selection are displayed when the participant answers, so that the participant really tries to recognize the regularities and does not only guess. For this we use the *Sandia Matrix Generation Software*, which was presented in the paper of Matzen et al. [24]. The starting point is the standardized *RPM* test. For us, it is particularly attractive because it is nonverbal, and thus the language center is not also inadvertently activated. If it is activated, it is because it is really needed. With this software

<sup>4</sup> see [https://answers.microsoft.com/de-de/windows/forum/windows\\_8-winapps/rechner-calculator-unter-windows-8-punkt-vor/c22d9994-bea3-46d2-87e4-545c7112b78c](https://answers.microsoft.com/de-de/windows/forum/windows_8-winapps/rechner-calculator-unter-windows-8-punkt-vor/c22d9994-bea3-46d2-87e4-545c7112b78c) , last visited: 15.02.2021



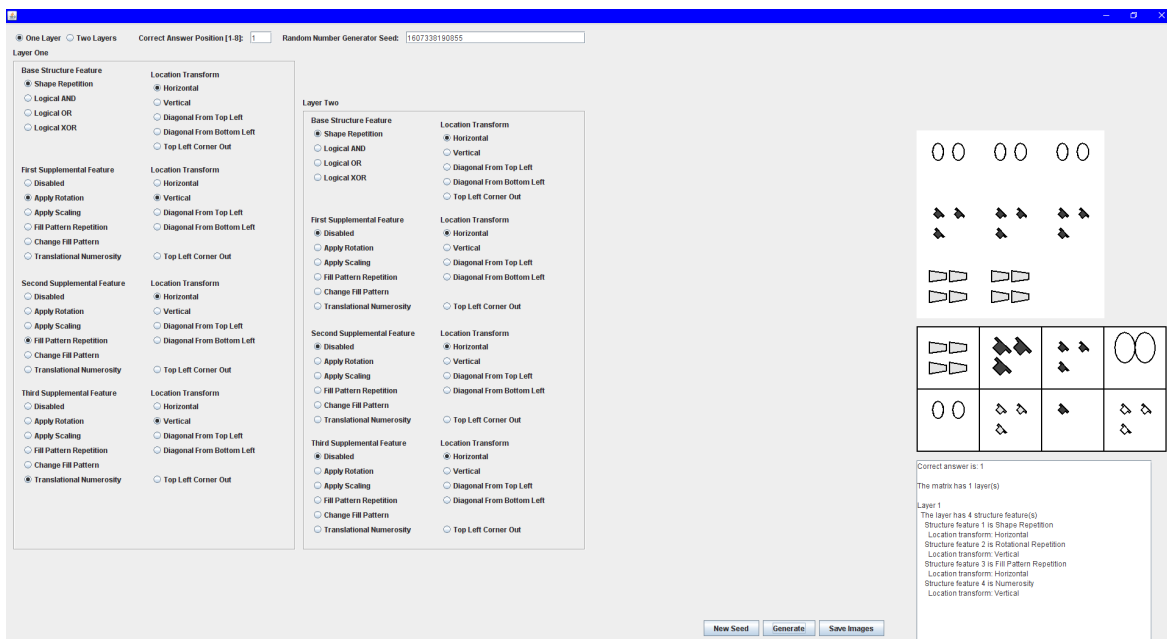


Figure 4.1: The graphical user interface of *Sandia Matrix Generation Software* [24]

we can create our own RPM, which according to the paper by Matzen et al. can be easy or difficult, depending on the desired difficulty. This software was developed primarily for neuroimaging studies to generate many matrix tests at the push of a button. The graphical interface (Figure 4.1) of the program allows to choose any transformation within the matrix itself. In the next section one will find an overview of the used transformations and an estimation of the difficulty based on the results of Matzen et al. We can choose between: logic problem (AND, OR, XOR) transformation and other transformations called *Three-Relations*, which changes the shape, shading, orientation, size and number of the shapes. The shapes used are ovals, rectangles, triangles, diamonds, trapezoids and T-shapes.

We distinguish between the difficulty levels of easy, medium, hard and special (Figure 4.2). According to the paper by Matzen et al. [24], the easy transformations are *Three-Relations* with shape repetition. All types of this transformations are equally difficult. The difficulty of the resulting task depends on the number of applied transformations. In our case we applied four transformations to every RPM. The directions horizontal and vertical are easier than diagonal, so we omitted diagonal here. Besides the *Three-Relations* there are also logical transformation, namely AND, OR, XOR. The correctness rate of OR is higher with 46,3% than AND and XOR, both 33,8%. However, the differences in their difficulty were not significant, which may be due to a small number of participants. In our case easy consists of shape repetition horizontal plus rotation horizontal plus numbers vertical plus shading vertical. The medium tasks are logical OR horizontal plus rotation vertical and the hard logical AND horizontal plus rotation vertical. The special tasks are not clearly placeable in this series. It is easier than hard, but based on the paper we cannot place it clearly, before or after medium. For the special tasks, logical XOR horizontal was applied.

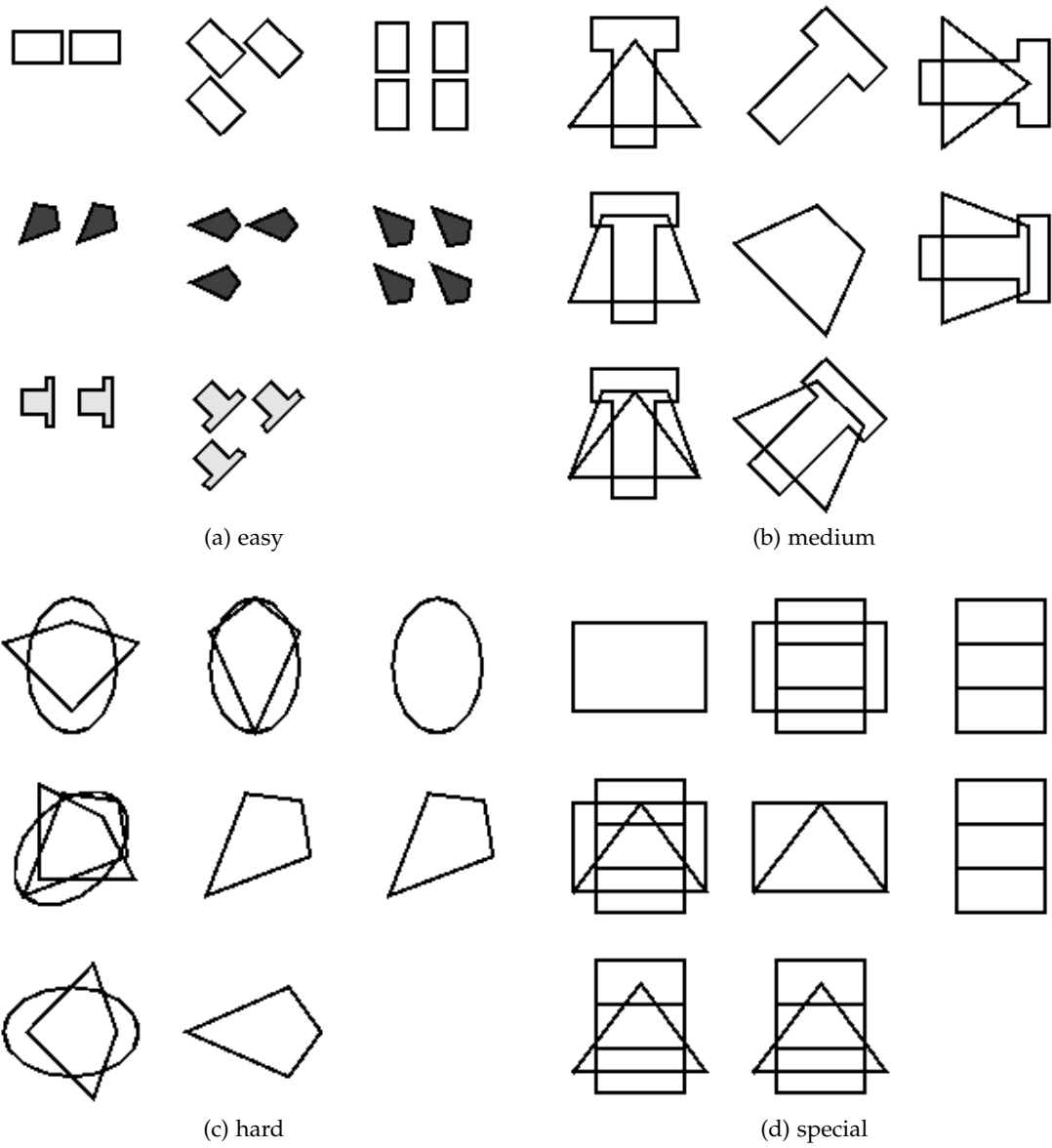


Figure 4.2: Example of four matrix tasks with different difficulty. The 9th symbol on the bottom right must be derived and is not shown.

#### 4.4 PROGRAM COMPREHENSION TASKS

In the Related Work in Chapter 3, we presented several studies on program comprehension. We mainly refer to the studies by Siegmund et al. [38] and Peitek et al. [27]. As in these studies, we use *Java* code snippets. However, in this work we do not ask directly what the program outputs for a given input, but instead only in the second part of the task. In this way the program has to be really understood. The functionality has to be abstracted and the participant has to remember it briefly. Then the participant sees the input and four possible outputs. Now they have to choose the right output. To summarize, the participant can see the program, then, when answering, the participant has to enter what output the program generates for the input that is displayed. The goal is that the functionality of the program is understood, such as adding the numbers or reversing the character string. Thus, the correct of the four solutions can be chosen very easily afterwards. The difficulty of the task is to understand the source code snippet, giving the output is only an encouragement and control mechanism, to determine if the participant has understood the code snippet. Variable names were chosen to be meaningless, for example, number1 and number2 instead of base and power. But result is always the variable whose content is returned (Listing 4.3). This way, we enforce bottom-up comprehension. In this study, we have reused 14 *Java* code snippets from the studies by Siegmund et al. [38] and Peitek et al. [27]. Six more *Java* code snippets were designed and added by us that were of similar nature. All of the code snippets are listed in the Appendix A.

Listing 4.3: Example of a program comprehension task

```
static int compute(int a, int b) {
    if (b == 0) {
        return 1;
    }

    if (b == 1) {
        return a;
    }

    return a * compute(a, b - 1);
}

// What does the program output if you enter a = 3 and b = 2?
// a) 1
// b) 6
// c) 8
// d) 9
```

## 4.5 VALIDITY

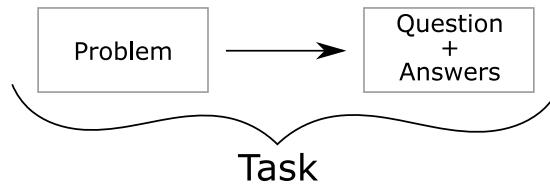


Figure 4.3: The structure of a task

We designed the tasks so that a few properties apply to all tasks. To ensure that the [fMRI](#) data is of high quality, we filter out the data where the participant did not understand the task or did not complete it correctly. Therefore, the procedure for each task is the same (Figure 4.3). First the problem definition is shown, this is the *problem part*. After that the question and the answer options are displayed. Now the participants should select the correct answer, this part is called *Q&A*. A similar approach is also used in the paper by Liu et al. [21]. If they do not select the correct answer, we must assume that they could not solve the task and thus the desired brain activities might did not take place.

## EXPERIMENT

---

This chapter describes the main experiment that is conducted for this thesis. It serves as a prestudy for the follow-up fMRI experiment outlined in Chapter 7.

### 5.1 PRESTUDY FOR THE FMRI STUDY (MAIN STUDY IN THIS WORK)

The prestudy has taken place online. The participants work on the tasks and the processing time, as well as the chosen answers are saved.

#### 5.1.1 Demographic Questions

The study was distributed via student email mailing lists. The student councils of several German universities<sup>5</sup> were contacted, but unfortunately there was no feedback whether the emails were forwarded to the students. A high number of responses came shortly after the email was sent via the student email distribution list of the department of computer science at the University of Saarland. The participants received no payment for this study.

Before the core part of the study began, the participants were asked a few demographic questions. These were, age, if they are studying a computer science related subject and if they already have a university degree. The programming experience in *Java* as well as in general was self-assessed and given on a 5-point *Likert scale*. Additionally, the years they have been programming *Java* or in general and their time at university. They were also asked to rate their English level according to the Common European Framework of Reference for Languages<sup>6</sup> (i.e., A, B1, B2, C1, C2). Of course, participants were asked for their consent to use their data for research purposes. After completing the survey, participants were asked if they enjoyed the study, if they were able to work undisturbed and if they understood the tasks or clicked on random answers. Finally, there was also an opportunity to express their ideas about our research topic in a free text field.

If participants contradicted the use of their data at the beginning or end of the study, it was excluded from analysis. The age structure and the field of study with degree helps us to better compare our study with other studies. The self-assessed programming experience with *Java* and in general as well as the years they have been programming helps us to exclude the data sets that do not show sufficient programming experience. In addition, we exclude the data sets that have a low level of English, since this study is conducted in

---

<sup>5</sup> Universität des Saarlandes, Karlsruher Institut für Technologie, Hochschule für Technik und Wirtschaft des Saarlandes, Humboldt-Universität zu Berlin, Christian-Albrechts-Universität zu Kiel, Technische Universität Kaiserslautern, Universität Paderborn, Rheinische Friedrich-Wilhelms-Universität Bonn, Goethe-Universität Frankfurt am Main

<sup>6</sup> <https://www.coe.int/en/web/common-european-framework-reference-languages>

English. The questions at the end of the study, about the meaningfulness of the answers and how often they were interrupted during the study also help us to exclude low-quality data sets. The question about fun and the free text answer does not directly help us with this study but serves as feedback to improve future studies.

### 5.1.2 Online Survey Tool

The survey tool *SoSciSurvey*<sup>7</sup> was used for this study. It allows scientific research for free and provides an extensive range of task types. Thus, in addition to the demographic data already mentioned, we were also able to measure the response times for the actual tasks. The collected data can be downloaded as a *Comma-Separated Values (CSV)* file.

### 5.1.3 Setup

In this study, 20 mathematical tasks, 20 language tasks, 20 RPM tasks and 20 programming tasks were tested for their suitability to be used in an fMRI study. Therefore, we need to understand whether they were answered correctly and also measure the time based on the correctly answered data sets. Since 80 tasks is too much for an online study, the online questionnaire divided participants into two groups, that is Team A and Team B, based on whether the day of their birthday is even or odd. When participants were finished with their own group, they were offered to complete the second group as well (Figure 5.2). Thus, these groups are not distinct groups in the sense of the experimental design as it is a within-subject design.

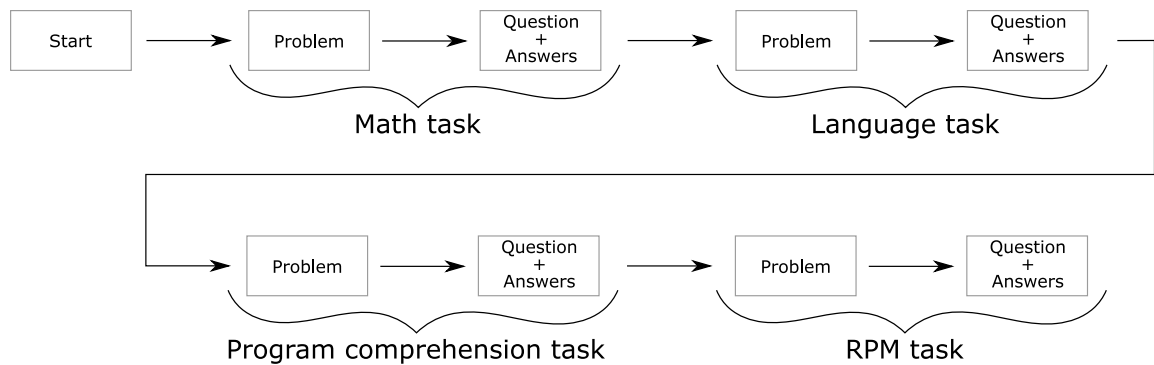


Figure 5.1: The structure of a block

The study consisted of two times 10 blocks. One block (Figure 5.1) consists of a mathematical task, a language task, a programming task, and a matrix task. An additional block was available as a sample block at the beginning of the study (Figure 5.2) to become familiar with the design. A block begins with the participant pressing start. This is to allow an individual pause after or before each block. Then the problem is displayed (Figure 5.3). When the participants have understood it, they can click on "Go to answer submission...".

<sup>7</sup> <https://www.sosicisurvey.de>, Version 3.2.21, last visited: 18.02.2021

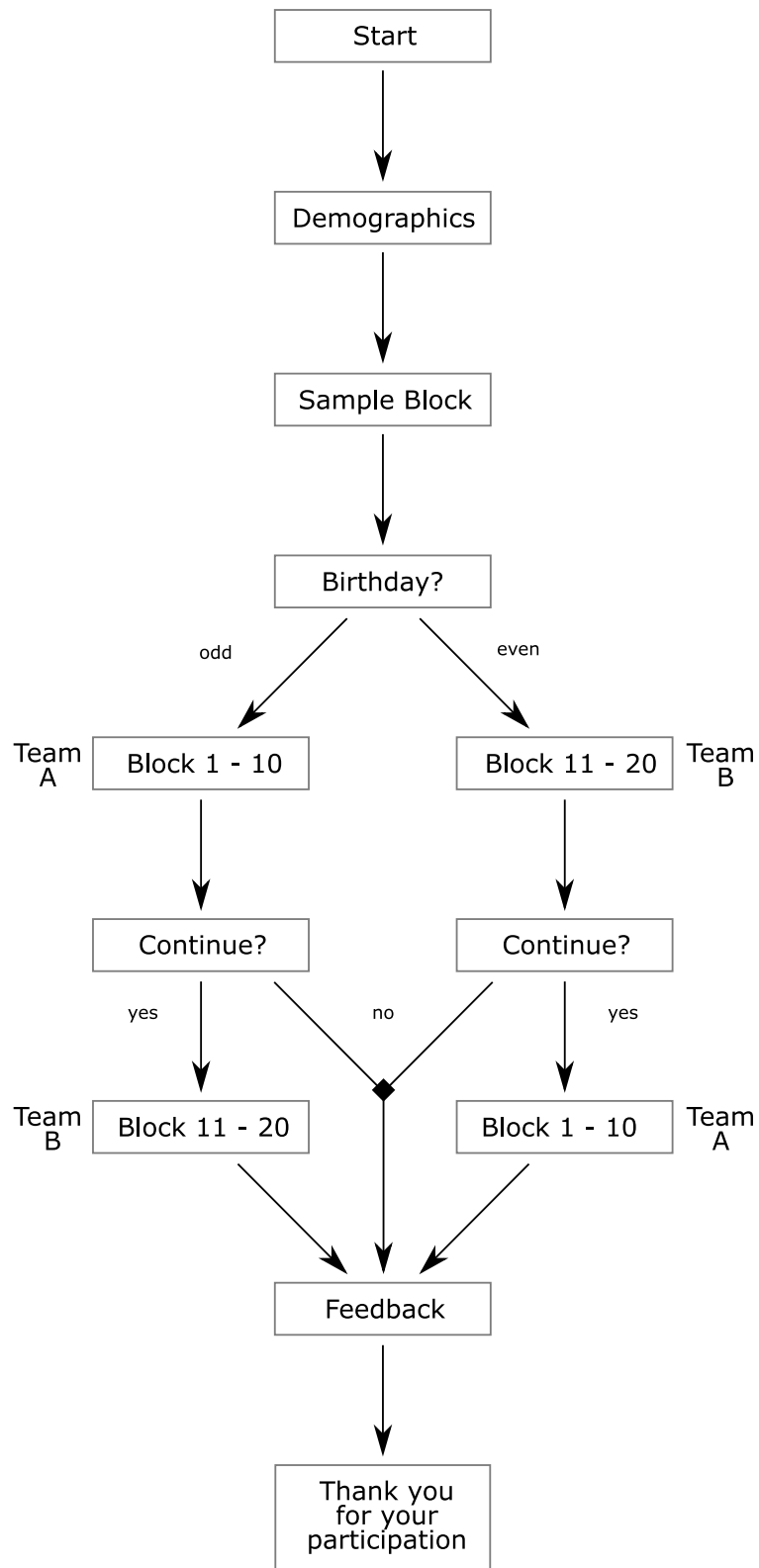


Figure 5.2: The structure of the online study

The time they need for this *problem part* is called *comprehension time*. On the next page there is no longer the task displayed, but four answer options with numbers and the additional answer options of "None of these", "I don't remember", and "I have no idea" (Figure 5.4). The time that elapses, during the *Q&A part*, until one of these options is clicked is called *answer time*. For the evaluations we will mainly deal with the *comprehension time*.

It was randomly determined at which position the correct answer was placed. The three additional answer options are always placed after the concrete answer in their fixed order. The correct answer was always a concrete answer. Thus, the choice "None of these" only had the sense that the participants would not guess an answer if their original answer was not available among the four answer options. The answer choice "I don't remember" helps us to understand if remembering the solution is a common problem. Participants may select the answer "I have no idea" if the task was not understood, they could not solve it or if the participant decided not to answer this question. This is also better than randomly choosing any of the 4 answer choices, because we can classify the different types of errors. This gives us a better overview which problem we have to solve before using the tasks in the *fMRI* study. This way we can be more sure that a chosen answer option was not guessed. In summary there are these three types: The participant's solution was not there; the participant could not remember the solution or the participant did not understand the question or does not want to answer.

#### 5.1.4 Screenshots of the online survey

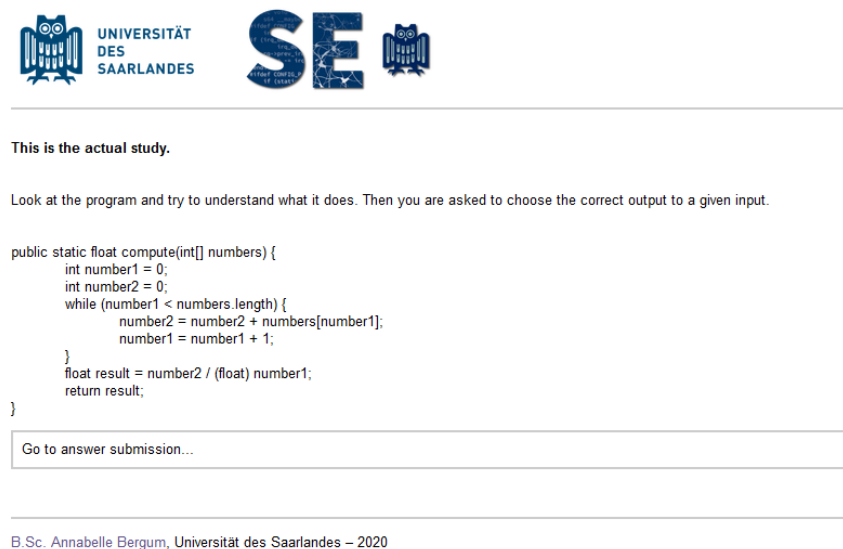
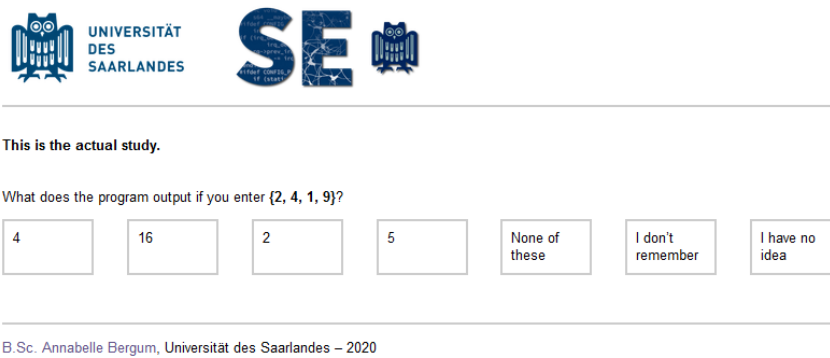


Figure 5.3: Screenshot of the online survey showing the *problem part* of the program comprehension task





UNIVERSITÄT  
DES  
SAARLANDES

SE

---

This is the actual study.

What does the program output if you enter {2, 4, 1, 9}?

4	16	2	5	None of these	I don't remember	I have no idea
---	----	---	---	---------------	------------------	----------------

---

B.Sc. Annabelle Bergum, Universität des Saarlandes – 2020

Figure 5.4: Screenshot of the online survey showing the *Q&A part* of the program comprehension task

## 5.2 RESEARCH QUESTIONS

The purpose of this preliminary study is to answer the following research questions:

**Research Question 1.** Are the tasks presented in Chapter 4 understandable? Will participants be able to solve them?

In an *fMRI* study, the participants can only work on a limited number of tasks so that the experiment does not last too long. Nevertheless, we can only use the brain data of the correctly answered tasks for the evaluation. Therefore, the participants should answer as many tasks as possible correctly. In the prestudy, we need to test whether the tasks are generally feasible and understandable, by assessing their correctness rate, before we use them in the *fMRI* device.

**Research Question 2.** Is the amount of time used to answer the tasks within an optimal length?

It is important that the time taken to complete a task in the *fMRI* is not too short. In this case, the *BOLD* signal does not build up sufficiently and we cannot reliably identify differences in brain activation. Additionally, the time taken to complete a task should not be too long. There is only a limited amount of time for the participant to complete all tasks in the scanner. During the limited experiment time, we aim at presenting as many tasks as possible to collect more data so that the analysis has an increased statistical power. Furthermore, if tasks are longer than 60 seconds, the errors become very large, so we should limit the length to 60 seconds anyway. So, in the prestudy we want to collect response time data to estimate the time the participant needs to solve a task in the *fMRI* scanner.

**Research Question 3.** Are the new and modified program comprehension tasks comparable to the program comprehension tasks used in former studies by Siegmund et al. [38] and Peitek et al. [27]?

In this study, we adapted the work of Siegmund et al. [38] and Peitek et al. [27]. There, 14 program comprehension tasks were used. However, in our study we needed 20, so we designed six new code snippets. In addition, a few of the original code snippets were slightly modified, to fit the general definitions of the intended algorithms. To compare the results of the study, we are interested in whether there are major differences between the original and the new code snippets.

**Research Question 4.** Is the difficulty of the mathematical, language and *RPM* tasks comparable to the difficulty of the program comprehension tasks?

With our approach, we develop a baseline for future *fMRI* studies of program comprehension. Therefore, we would like to adjust the level of difficulty of the baseline tasks to that of the program comprehension to enable a better comparison.

## EVALUATION

---

### 6.1 EVALUATION PIPELINE

First, we present our evaluation pipeline. With the help of these successive steps, we can prepare the data and subsequently evaluate it.

#### 6.1.1 Technical

The online survey website provides the possibility to download the data sets as a [CSV](#) file. This file is now processed by the evaluation pipeline, which is implemented in *Python* (version 3.7.6) in a *Jupyter Notebook* (version 1.2.6). The libraries we use are *pandas* (version 1.0.1) for data processing, *matplotlib* (version 3.1.3) for plots, *numpy* (version 1.18.1) for mathematical calculations and *scipy.stats* (version 1.4.1) for the *Mann-Whitney U test* to determine statistical significance.

#### 6.1.2 Clean Data

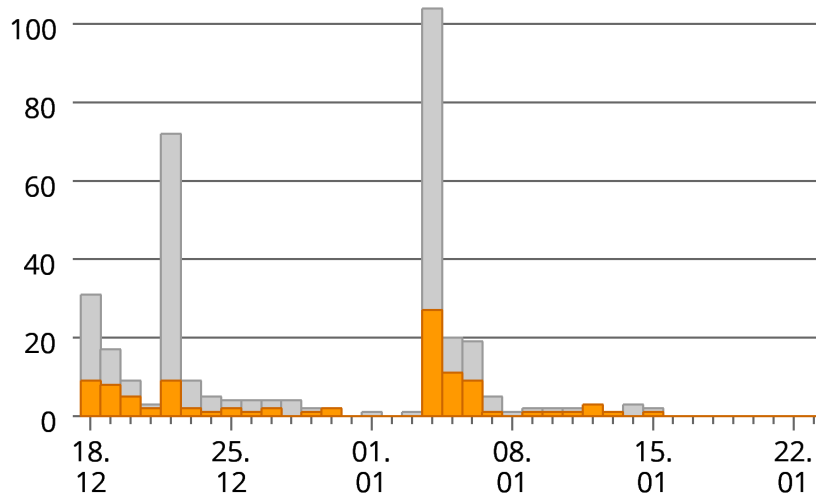


Figure 6.1: The participation within the test period (18.12.2020 to 18.01.2021). All clicks on this survey are shown in gray, the orange subset shows the number of questionnaires that were processed until the end.

**DEMOGRAPHICS** The survey began with a questionnaire asking for the demographic data of the participants. These data are used to make the study reliable and to identify the lack of qualifications to participate in a program comprehension study. We only evaluate the data of the participants, who meet these following criteria.

- The questionnaire was completed during the test period (Figure 6.1).
- The participant must give consent for the collected data to be analyzed for our research purposes.
- The participant declares to have at least the B1 level of English.
- The participant states to have at least one year of programming experience.
- The participant rates their programming experience as at least *not so good* on a scale of *rather poor, not so good, mediocre, good, very good*.
- The participant claims to have taken the questionnaire seriously and not just clicked through for fun.
- The participant should not have been distracted too often. Thereby we discard a data set at the statement *often distracted* on a scale of *full attention, once distracted, often distracted*.
- The participant scores their own responses as *everything meaningful*, on a scale of *everything meaningful, sometimes not meaningful, often not meaningful*.

The data sets, which do not fulfill these criteria, are excluded from the further steps. In our case, 19 were excluded because they were created for debugging purposes before the start of the study, another 16, because they lacked programming experience and another 11 because they lacked consent, were often distracted or clicked not meaningful answers multiple times. Thus, a total of 46 out of 351 data sets were removed. This leaves 305 data sets for further processing.

**INSUFFICIENT DATA** The file downloaded from the survey website also contained records where only the link was clicked or only demographics and or the sample task was done. Therefore, the next step is to process the data sets that did not have any task (exclusive sample task) completed, neither in the first nor in the second half of the study. These data sets were removed before further processing. There were 166 of 305 unprocessed records in the task category. This results in 139 data sets remaining for the next steps. Team A has 87 and Team B has 96 records, that are not completely empty. Here it is noticeable that 87 plus 96 would be 183 data sets, which means that 44 participants have participated in both groups.

**UNUSUAL PERFORMANCES** Since this was an online study and not a face-to-face study, we cannot be sure if the study was conducted seriously. Similarly, we are also biased towards the reliability of the participants in the demographic data. Therefore, we filter out additional data sets that either contain a large number of incorrect answers, or that exceptionally often took much longer to answer than the other participants.

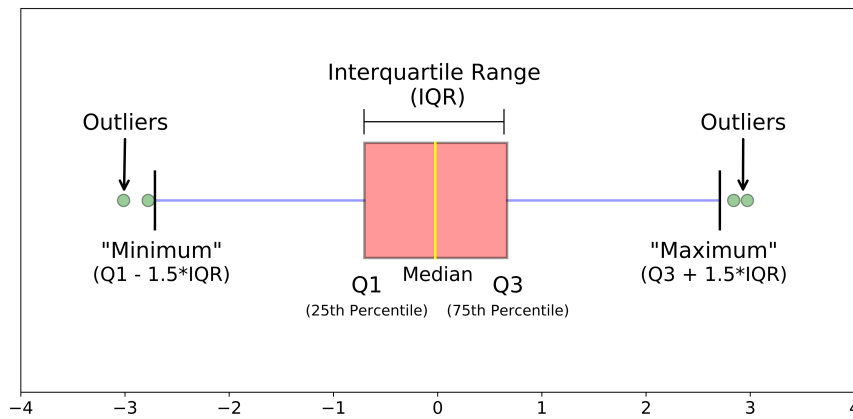


Figure 6.2: A visual explanation of the structure of the boxplot with outliers.<sup>8</sup>

**BACKGROUND: OUTLIER** We calculate the outliers (Figure 6.2) as they are computed in the *matplotlib* boxplots. The documentation of the boxplot states that the *interquartile range (IQR)* is the range between the first and the third quartiles. The lower whisker is calculated by subtracting  $1.5 \cdot IQR$  from the first quartile. Analogously the upper whisker is calculated by adding  $1.5 \cdot IQR$  to the third quartile. Everything beyond the range of the whiskers is considered as outlier.<sup>9</sup>

Thus, we calculate the 3 quartiles. Then we calculate the interquartile distance by subtracting the edge of the third quartile from the edge of the first quartile. So, anything less than 1.5 times the interquartile range below the first quartile is an outlier. As well as anything above 1.5 times the interquartile range above the third quartile is an outlier.

<sup>8</sup> <https://towardsdatascience.com/understanding-boxplots-5e2df7bcd51>, last visited:19.02.2021

<sup>9</sup> The documentation of the *matplotlib*.boxplot whiskers available online at [https://matplotlib.org/3.1.1/api/\\_as\\_gen/matplotlib.pyplot.boxplot.html](https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.boxplot.html) last visited on 18.01.2021.

**TOO LOW CORRECT ANSWER RATE** In the next step, we would like to exclude the participants who have an exceptionally low rate of correct answers. Here, the correct answers and the answered questions are counted, since some of the participants did not answer the full 10 blocks, but for example only 8. In order not to distort the correctness rate, the unanswered questions are not counted in the total number of the individual participant. After all correctness rates have been calculated, we let the outlier of these values be determined. We discarded 8 out of 139 data sets because the correctness rate was too low, the accepted minimum is 72% (Figure 6.3). This yields 131 data sets in the following evaluations.

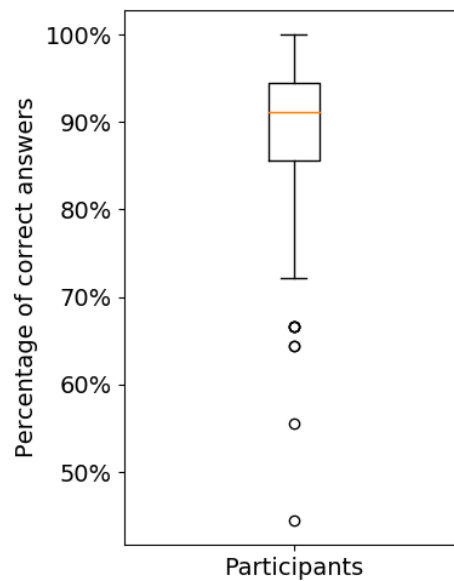


Figure 6.3: The distribution of the correct answer rates among participants.

**UNUSUAL RESPONSE TIME** Answering a simple question after understanding the task serves as an assurance that the task has truly been completed and understood. Thus, we only include the *comprehension* times of correctly answered tasks in our analysis. Recall *comprehension* time means the time the participant saw the task, not the time it took to select the correct answer. There are outliers in the *comprehension* times (Figure 6.4), possibly the participants were interrupted or distracted. The average outlier rate per participant is: 7%. Consequently, in the data set of the outlier rate per participant the rates greater than 20% are outliers (Figure 6.5). We did not utilize these participant data sets any further. These comprised 11 of 131 data sets. Thus, 120 data sets remain for the analyses.

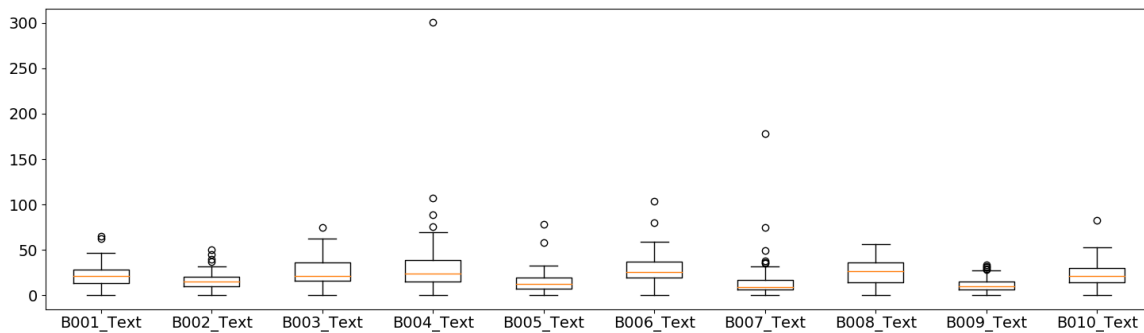


Figure 6.4: The exemplary distribution of the *comprehension* times per task.

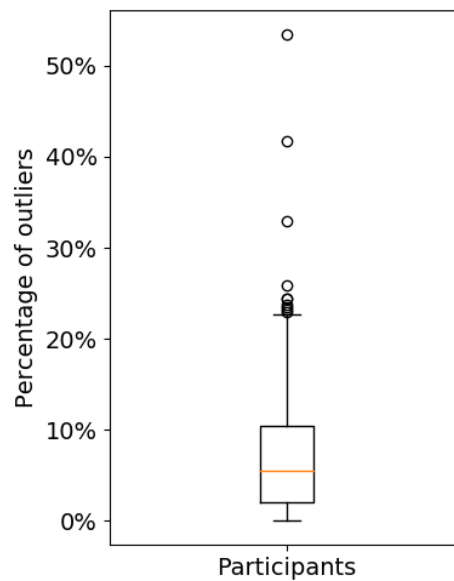


Figure 6.5: The distribution of the outlier rate per participant.

**SUMMARY** In this way, the collected data is cleaned (Figure 6.6). We can now start with the analysis.

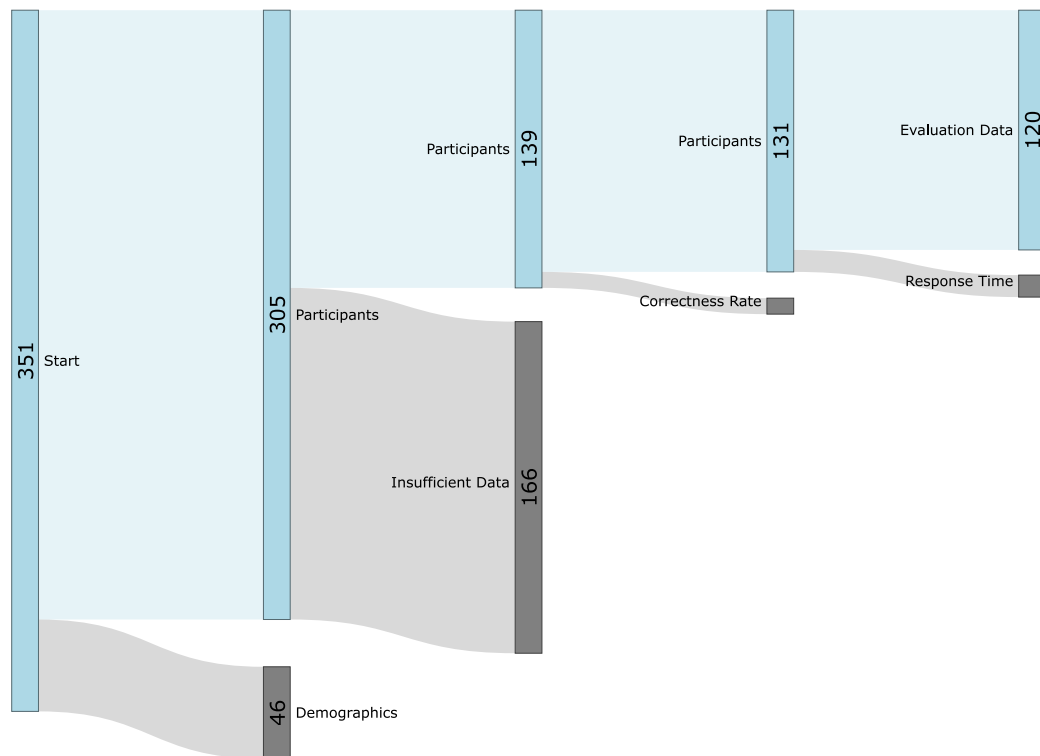


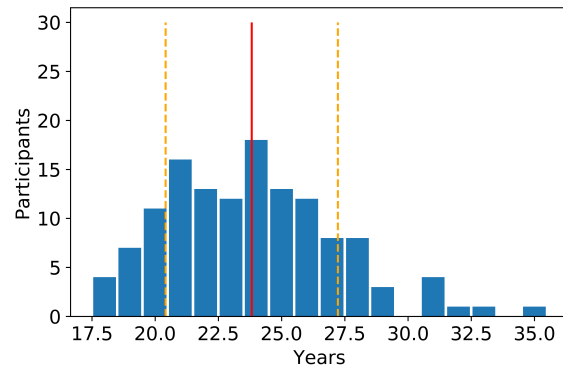
Figure 6.6: The amount of data points which are discarded through the cleaning of the data.<sup>10</sup>The Sankey Diagram Generator by Dénes Csala based on the Sankey plugin for D3 by Mike Bostock (2014), <https://sankey.csaladen.es> last visited: 22.01.2021

### 6.1.3 Participant Demographics

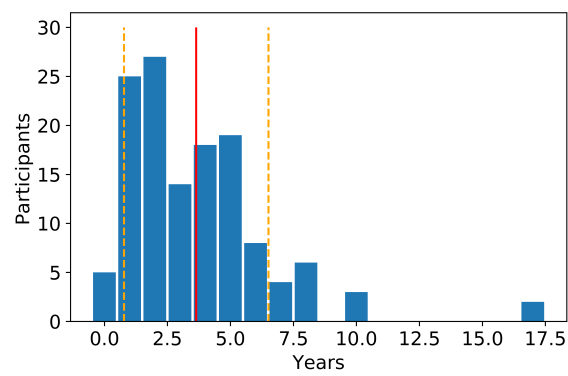
After we have cleaned the data, we can now give the demographics of the study. 106 of the 120 participants come from a computer science-related course of study. The remaining 14 are divided into four biometrics, two computational linguistics and a few embedded systems, mathematics and several engineering courses as well as business administration and economics. The average age is 23.7 years ( $\pm 3.42$  years). The average time at university is 3.58 years ( $\pm 2.88$  years), while the programming experience is slightly longer, averaging 5.54 years ( $\pm 3.72$  years). The programming experience with the *Java* language is again similar to the study time with an average of 3.21 years ( $\pm 3.04$  years) (Figure 6.7). 56 of the participants are in their undergraduate program, 46 further have a bachelor's degree, 16 have a master's degree, two have a PhD. Programming experience is classified as good by 55.83% and very good by another 14.17%. 70.83% of the participants have an English level of C1 or C2, the remaining 29.17% have B1 or B2 (Figure 6.8). The detailed numbers can be found in the Appendix A.

<sup>10</sup> The Sankey Diagram Generator by Dénes Csala based on the Sankey plugin for D3 by Mike Bostock (2014), <https://sankey.csaladen.es> last visited: 22.01.2021

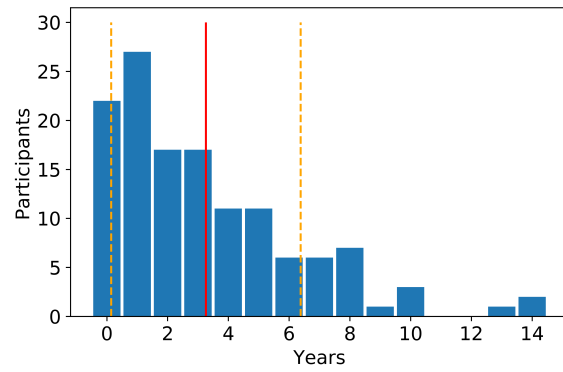
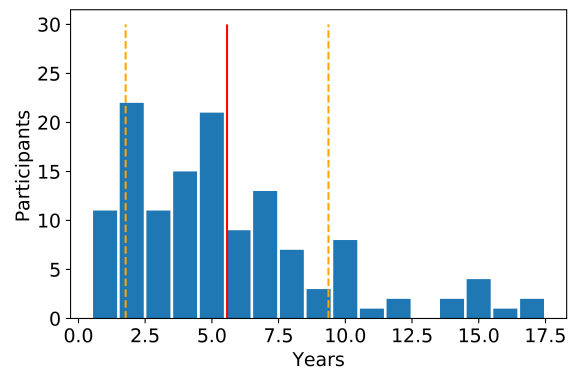




(a) Age Distribution

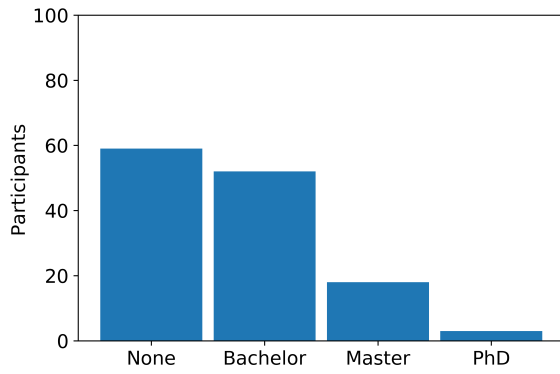


(b) Years at University Distribution

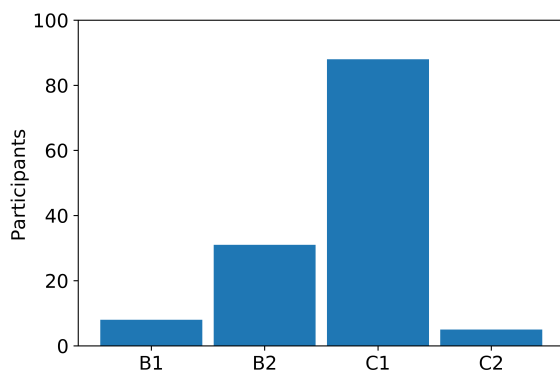
(c) Years of *Java* Experience Distribution

(d) Years of General Programming Experience Distribution

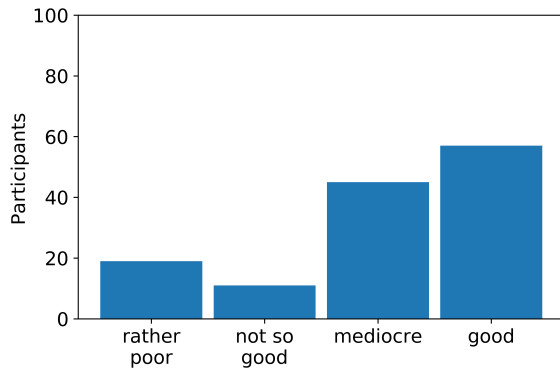
Figure 6.7: Overview over the age and years at university, programming and *Java* programming



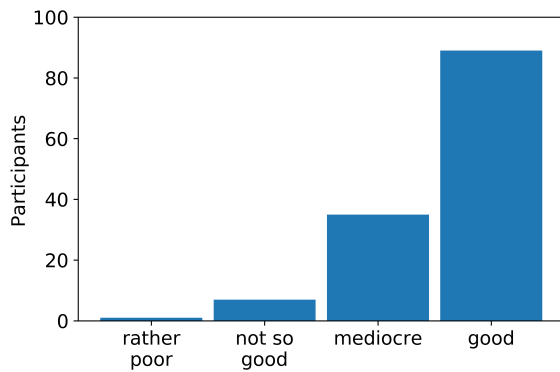
(a) Degree



(b) Rating of English Level



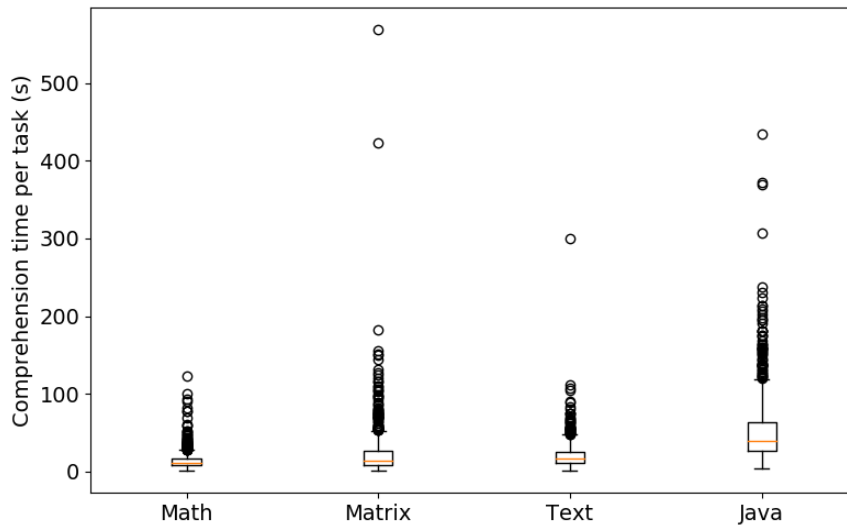
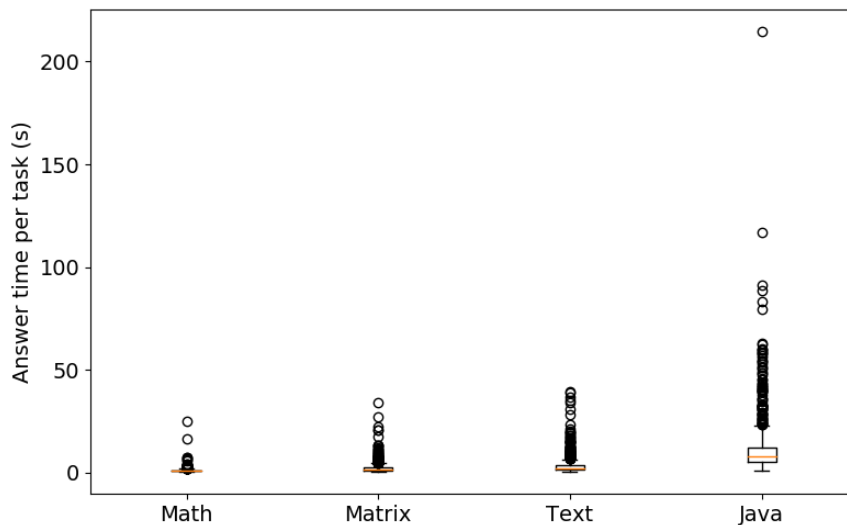
(c) Rating of *Java* Experience



(d) Rating of Programming Experience

Figure 6.8: Overview over the qualifications of our participants

## 6.1.4 Response Times

Figure 6.9: The distribution of the *comprehension* times per task type.Figure 6.10: The distribution of the *answer times* per task type.

We visualized the *answer times* for each category as a boxplot. Note that, for example, *Math* represents the time the participant was able to read the task until they clicked on "Go to answer submission" (Figure 6.9). The time *Math Submission* describes the *answer time* used to select the correct answer from the answer choices. We can see very strongly that the *answer time* is significantly higher for *Java* (Figure 6.10). This is due to the fact that in the *Java* task the submission of the answer was not only a search for the correct solution, but also the producing of the output to the now only visible input. With the three others, only the correct solution had to be shortly memorized and then clicked on. With *Java* one had to remember the functionality of the function and then compute the output of the function according to the given input and then click on the output in the answer options.

**BACKGROUND: SIGNIFICANCE TESTS** The *Mann-Whitney U test* is used to investigate central tendencies of two independent samples. The test has very low requirements on the test data. The values have to be scaled ordinally, like the *Likert scale* or seconds in time. Our data is ordinally scaled, as the values have an order as well as the distance between them is linear scaled. This applies also to the correctness rate, these are numbers between 0 and 1. The *Mann-Whitney U test* is therefore not be disturbed by outliers, like other tests based on normal distributions [11].

In the significance analysis the following order for the *answer time* came up: *Math* < *Matrix* < *Text* < *Java* (Figure 6.11). For the *comprehension time* the same scheme is available (Figure 6.12). The response distributions are to some extent normally distributed (Figure 6.13). All the distributions of the *answer times* are shown in the Appendix A.

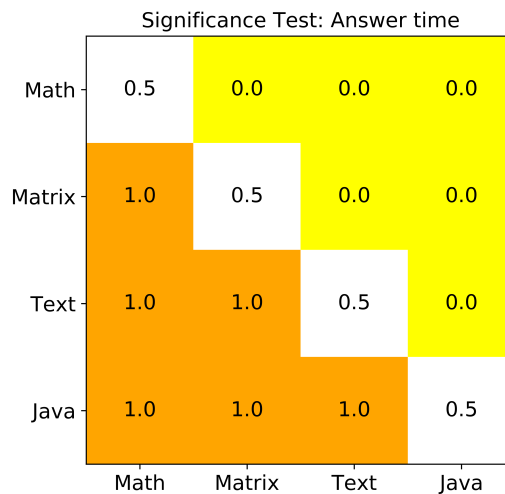


Figure 6.11: The p-values of the significance test of all combinations of the *answer times* (rounded to 4 decimal places). It reads as follows: Test <y-axis> less than <x-axis> had an p-value =.... for example, *Math* < *Text* has p-value = 0.0. (significance level  $p = 5\%$  and  $n = 20$ )

Significance Test: Comprehension time

Math	0.5	0.0	0.0	0.0
Matrix	1.0	0.5	0.0019	0.0
Text	1.0	0.9981	0.5	0.0
Java	1.0	1.0	1.0	0.5
	Math	Matrix	Text	Java

Figure 6.12: The p-values of the significance test of all combinations of the *comprehension* times (rounded to 4 decimal places). (significance level  $p = 5\%$  and  $n = 20$ )

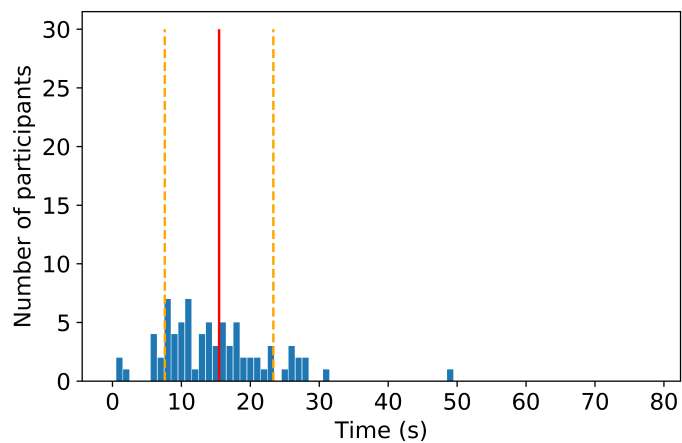
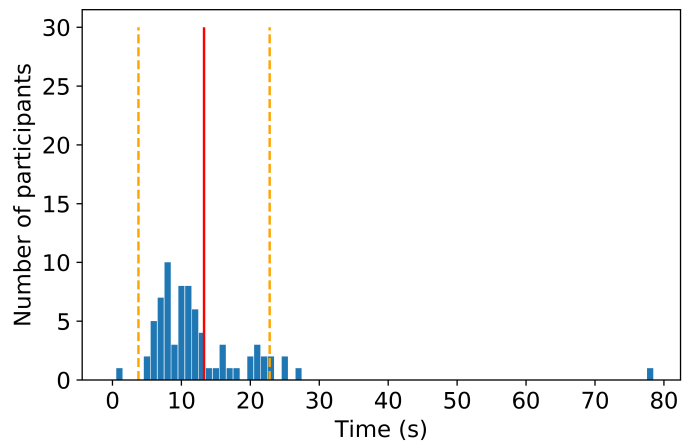


Figure 6.13: The distribution of the *comprehension* time in two single task, further histoplots are available in the Appendix A.

6.1.5 Correctness Rate

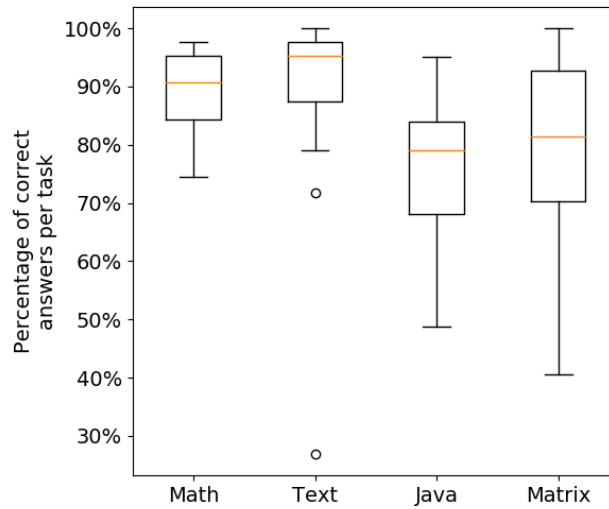


Figure 6.14: The distribution of the correctness rate per task type.

For each task, the correctness rate was determined and then grouped by type (Figure 6.14). The two outliers for *Text* (Bo03 "security system" and Bo05 "windmill") will be excluded for further research. Furthermore, the correctness rates for *Java* are significantly smaller than those for *Text* ( $p=0.0002$ ,  $n = 20$ ) at the 5% significance level. The correctness rates for *Java* are also smaller than those for *Math* ( $p=0.0002$ ,  $n = 20$ ). And those of the *Matrix* tasks are smaller than *Text* ( $p=0.0121$ ,  $n = 20$ ) and *Math* ( $p=0.0369$ ,  $n = 20$ ). This results in the following ranking order *Java, Matrix* < *Text* and *Java, Matrix* < *Math*. All other differences are not significant (Figure 6.15).

Math	0.5054	0.0914	0.9745	0.9998
Text	0.913	0.5054	0.9909	0.9999
Matrix	0.0272	0.0098	0.5054	0.7879
Java	0.0002	0.0001	0.22	0.5054
	Math	Text	Matrix	Java

Figure 6.15: The p-values of the significance test of all combinations of the correctness rate (rounded to 4 decimal places). (significance level  $p = 5\%$  and  $n = 20$ )

In the distribution of answers per task type (Figure 6.16, 6.17, 6.18, and 6.19) it is noticeable that most of the tasks were answered correctly. In mathematics, the selection option "None of these" was used more often than there were incorrect answers. Thus, it fulfilled its purpose. The participants acknowledged honestly that their solution did not appear in the 4 possible answers. For program understanding, "I don't remember" was chosen more often than for the other task types. But it was still a small amount. Therefore, we can assume that remembering the problem just seen, did not have a negative influence on the survey. Of course, some tasks will be omitted because the participants cannot remember, but this mechanism offers a nice opportunity to check the understanding of the task. The selection "I have no idea" was chosen most frequently of all task types in the RPM. Either the participants did not want to complete this task or the participants could not solve these tasks. This is also a numerically low proportion. Overall, we have a very high correctness rate and the wrong answers cannot be ascribed to a bad approach of our study, otherwise the generic answer options would have been chosen more often.

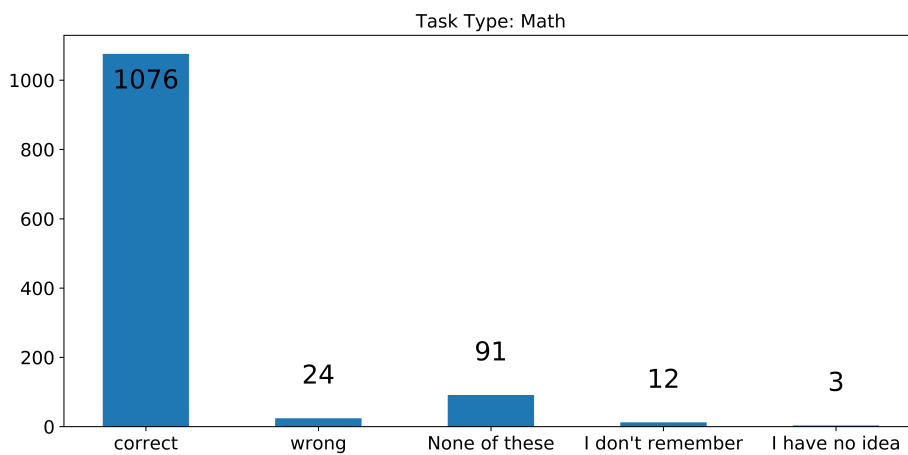


Figure 6.16: The distribution of the selected answer options in the mathematical tasks.

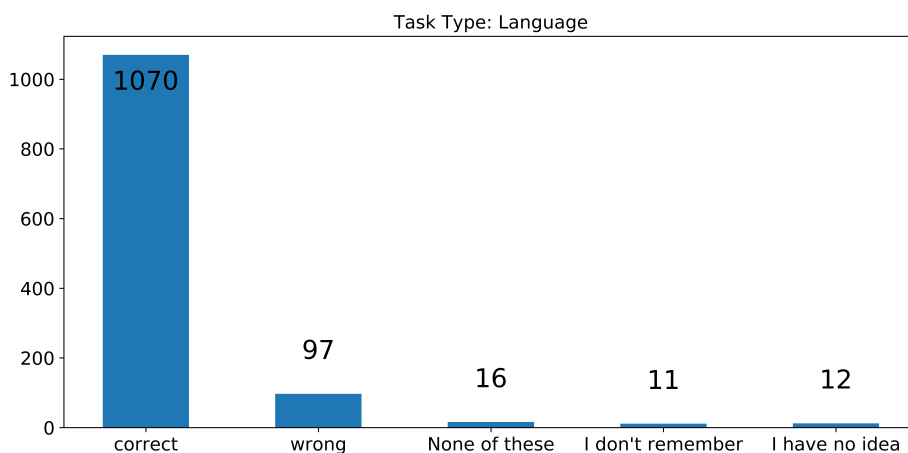


Figure 6.17: The distribution of the selected answer options in the language-oriented tasks.

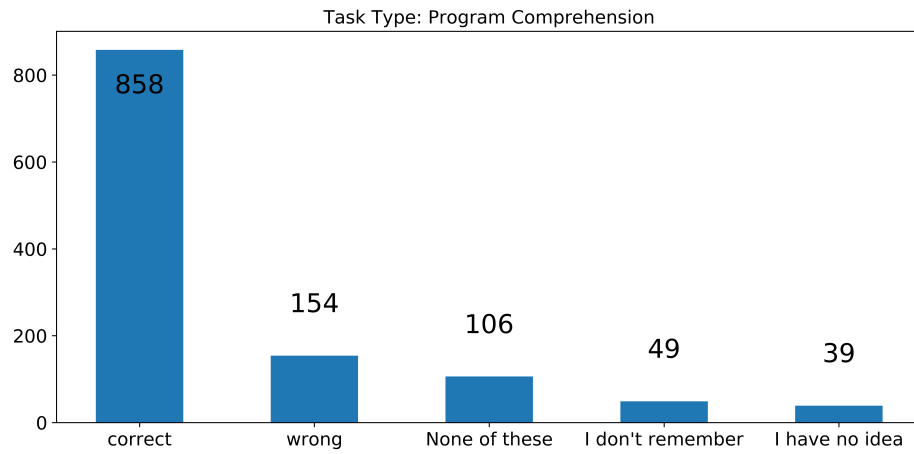


Figure 6.18: The distribution of the selected answer options in the program comprehension tasks.

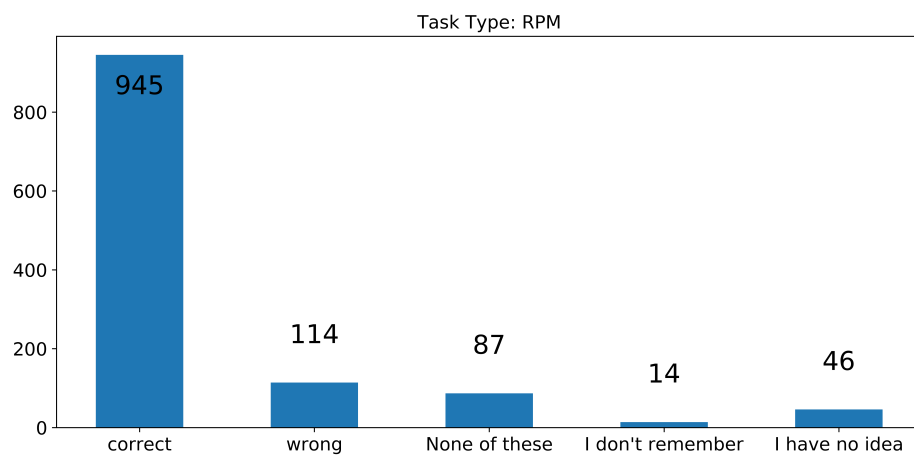


Figure 6.19: The distribution of the selected answer options in the [RPM](#).



### 6.1.6 Original vs. New Program Comprehension Tasks

Furthermore, we are interested in the comparability of the original code snippets and the new designed code snippets. We used 14 code snippets from the studies by Siegmund et al. [38] and Peitek et al. [27] and added 6 additional code snippets. The snippets that were only slightly modified are counted as original, and the completely redesigned ones are counted as new. So, we have 12 original and 8 new code snippets. We compare the distribution of the correctness rate and the *comprehension time* (Figure 6.20). The original code snippets have a significant higher mean correctness rate 81% (standard derivation: 0.12) than the new code snippets with a mean of 70% (standard derivation: 0.09) (Figure 6.21). The *comprehension time* of the new code snippets with 54.58 *seconds* (standard derivation: 19.93) is slightly higher than that of the original code snippets with a mean of 52.51 *seconds* (standard derivation: 28.56), but it is not significant. Overall, the workload of the task is similar, but due to the low number of tasks, the median is not so expressive, but the individual times and correctness rates per task should be considered individually (Table 6.1).

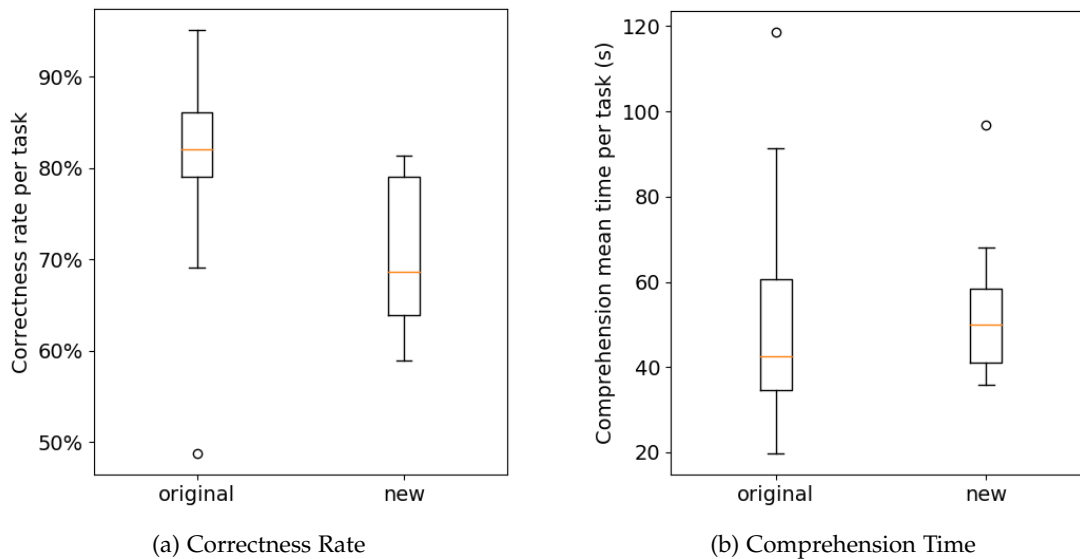


Figure 6.20: The original code snippets were compared to the new code snippets, both used in the prestudy in program comprehension tasks.

Task	Mean Time (s)	correctness rate (%)	original
CountVowels	32.70	95.12%	■
Palindrome	43.53	92.31%	■
Factorial	19.17	90.70%	■
Hurricane	35.16	84.62%	■
YesNo	29.65	84.62%	■
ArrayAverage	48.41	83.72%	■
Fibonacci	36.99	81.40%	
DumbSort	72.86	80.49%	■
CrossSum	40.16	79.49%	■
Power	42.90	79.07%	■
SquareRoot	50.07	79.07%	■
Prime	52.14	79.07%	
BinaryToDecimal	57.40	79.07%	
ForwardBackward	50.26	72.09%	
ContainSubstring	118.63	69.05%	■
LCM	61.31	65.12%	
Fast	93.19	65.12%	
EverySecondChar	52.94	60.47%	
AlwaysZero	38.65	58.97%	
CountLastWord	91.27	48.72%	■

Table 6.1: The mean *comprehension times* in seconds and the correctness rate of each snippet. The snippets are sorted in descending correctness rate. The ■ indicates, that this code snippet is an original one.

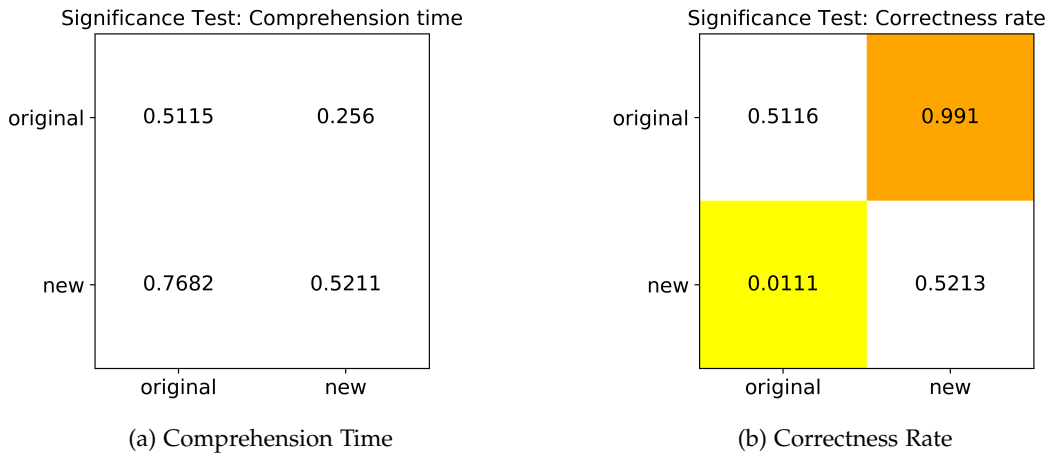


Figure 6.21: The p-values of the significance test of between the original and new program comprehension tasks of the *comprehension times* (rounded to 4 decimal places). (significance level  $p = 5\%$ ,  $n = 8$  and  $12$ , respectively)

### 6.1.7 Performance Prediction with Elastic Net

**BACKGROUND: ELASTIC NET** *Elastic Net* is a predictor algorithm which outperform other similar ones like lasso most of the time. Using linear regression, it generates coefficient for each input, so that the given output can be predicted. While during this the error square should be as small as possible [47]. In our case, we use the library *scikit-learn*<sup>11</sup>. Thereby we use 75% training data and 25% test data.

**BACKGROUND:  $r^2$  TEST** The used *scikit-learn* library has the possibility to calculate the  $r^2$  value of the predictions directly with test data. The best possible value is 1. Although the square in the name of the test suggests that 0 is the worst value, the value can be negative. This means that a bad model was generated in the first place.<sup>12</sup>

Finally, we try to evaluate if the performance of the program comprehension tasks can be predicted based on the performance data of the mathematical, language-oriented and RPM tasks. For this purpose, we use *Elastic Net* and the  $r^2$  test. Since *Elastic Net* delivers different values depending on the configuration of the random, we ran *Elastic Net* 50000 times. For predicting the *comprehension time*, the  $r^2$  value was not consistently greater than 0.4. A minimum of 0.5 would have had to be reached in order to make reliable assumptions. Therefore, the results only serve as a guide for further experiments. The best runs ( $r^2 > 0.3$ ) were saved (Table 6.2). The weights of the parameters of these runs were averaged. As a

<sup>11</sup> [https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.ElasticNet.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html), last visited: 08.03.2021

<sup>12</sup> [https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.ElasticNet.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html), last visited: 08.03.2021

result, the factor 0.42 was calculated for *Math*, the factor 0.08 for *Text* and the factor 0.80 for *Matrix*.

$r^2$	Math	Language	RPM
0.538	0.770	0.130	0.658
0.520	0.877	0.122	0.652
0.519	0.767	0.129	0.646
0.510	0.431	0.134	0.714
0.506	0.737	0.177	0.705
0.506	0.274	0.181	0.801
0.503	0.844	0.173	0.680
0.501	0.542	0.091	0.772
0.495	0.632	0.203	0.766
0.491	0.376	0.096	0.692
<b>Mean</b>			
0.340	0.423	0.079	0.799

Table 6.2: These are the 10 most successful runs of *Elastic Net* where  $r^2 > 0.3$ . These 10 out of 2995 data are from 50000 runs of the algorithm. For calculating the mean all data sets with  $r^2 > 0.3$  were taken into account.

$r^2$	CS subject	Age	Java Exp.	Program Exp.	Years at university	Years of programming	Years of Java	English Level
0.130	-1.068	0.523	-1.645	-2.307	0.000	-0.307	0.436	0.000
0.127	-1.148	0.755	-1.223	-1.605	-0.302	-0.148	0.218	0.000
0.124	-1.376	0.570	-1.619	-0.728	-0.128	-0.391	0.571	0.000
0.121	-2.521	0.582	-1.790	-2.058	-0.000	-0.255	0.478	0.123
0.116	-1.113	0.699	-1.404	-1.458	-0.065	-0.379	0.489	0.000
0.108	-1.702	0.663	-1.558	-1.681	0.103	-0.581	0.422	0.152
0.108	-1.926	0.635	-0.910	-1.500	-0.000	-0.387	0.526	-0.000
0.107	-1.103	0.651	-0.800	-0.965	0.067	-0.391	0.491	0.000
0.107	-1.133	0.691	-0.890	-1.534	-0.391	-0.168	0.170	0.000
0.106	-1.315	0.612	-1.857	-1.787	-0.265	-0.293	0.726	0.000
<b>Mean</b>								
0.109	-1.502	0.618	-1.501	-1.486	-0.036	-0.402	0.512	0.073

Table 6.3: These are the 10 most successful runs of *Elastic Net* where  $r^2 > 0.1$ . These 10 out of 20 data are from 50000 runs of the algorithm. For calculating the mean all data sets with  $r^2 > 0.1$  were taken into account.

We also tried to predict the *comprehension time* of the program comprehension tasks using the demographics information, but this was fairly unsuccessful. The  $r^2$  values were not greater than 0.2 with an average of 0.11 (Table 6.3).

### 6.1.8 Research Questions

After having presented our data, we now answer our research questions.

**Research Question 1.** Are the tasks presented in Chapter 4 understandable? Will participants be able to solve them?

The presented tasks have a high mean correctness rate of 91% for mathematical, 95% for language, 81% for RPM and 79% for program comprehension tasks. Therefore, most of the runs can be used for evaluation purposes. In Section 7.2, the correctness rates of each task can be seen. Consequently, we can answer this question in the affirmative.

**Research Question 2.** Is the amount of time used to answer the tasks not too short and not too long?

The tasks have an average *comprehension time* of 13.52 *seconds* for mathematical, 20.30 *seconds* for language-oriented, 19.80 *seconds* for RPM and 49.77 *seconds* for program comprehension tasks (Figure 6.9). The *comprehension time* of the mathematical tasks is relatively short, depending on the specific task, the BOLD signal may not be fully detected. The RPM and language-oriented tasks have a good duration, so that they do not require too much time in the scanner, but the BOLD signal should be well recognizable. As expected, the *comprehension time* for the program comprehension tasks is longer. However, this is still within acceptable limits overall. Only a few code snippets exceed the time of 60 *seconds* (Table 6.1). In Section 7.2 we discuss the required minimal and maximal length of a task in more detail. Furthermore, we there present the mean *comprehension time* of each task as well as the quartiles and the maximum time.

**Research Question 3.** Are the new and modified program comprehension tasks comparable to the program comprehension tasks used in former studies by Siegmund et al. [38] and Peitek et al. [27]?

The correctness rate and *comprehension time* of the newly developed or modified snippets have been compared to the original tasks from the study by Siegmund et al. [38] and Peitek et al. [27] in Section 6.1.6. The mean *comprehension time* is not significantly different, but the correctness rate of the new tasks is significantly lower than of the original tasks. Nevertheless, their correctness rate is very high, 70%. In the fMRI study the best out of both groups will be selected, regardless if they are original or new, to get the best possible tasks for the fMRI study.

**Research Question 4.** Is the difficulty of the mathematical, language-oriented and RPM tasks comparable to the difficulty of the program comprehension tasks?

The correctness of the task types ranges from 81% for RPM to 91% for mathematical to 95% for language-oriented tasks. The correctness for the program comprehension tasks is 79%. The correctness rates are quite high for all tasks types. The *comprehension times* are 13.52 *seconds* for mathematical, 20.30 *seconds* for language-oriented, 19.80 *seconds* for RPM and 49.77 *seconds* for program comprehension tasks (Figure 6.9). In this case, the

more difficult mathematical problems should be selected to bring the *comprehension time* in line with that of the [RPM](#) and language-oriented tasks. Likewise, the longest [RPM](#) and language-oriented tasks should not be used. A similar *comprehension time* for the program comprehension tasks is not possible to achieve, this should be prioritized on a high rate of correctness (Figure [6.14](#)).

## 6.2 THREATS TO VALIDITY

We divide the limitations of the presented experiment setup into internal and external validity and deal with them individually. Internal validity considers the clearness of the measurement in the experiment. Here it is important to avoid disturbing influences. Thus, only the independent variables are changed and all effects on the dependent variables can be traced back directly. This includes the tasks, the participants and especially the number of participants. External validity refers to the generalization of the results, referring at reproducibility and usage at work [\[40\]](#).

**INTERNAL VALIDITY** We have 120 participants for our prestudy, which is reasonably large. Unfortunately, we cannot guarantee more than presented in this chapter that the participants have filled out the questionnaire correctly and conscientiously. Furthermore, no follow-up questions could be asked to the study instructor if the instructions were not well understood. Furthermore, we did not use the original [RPM](#) from Raven [\[33\]](#) but, the self-generated ones with the presented tool by Matzen et al. [\[24\]](#). For the code snippets it may have happened that the algorithm was recognized without doing bottom-up comprehension.

**EXTERNAL VALIDITY** We used relatively short code snippets, like this they are not used in the everyday working environment. In addition, there are comments and helpful variable names, although not consistently. We have omitted and obfuscated these here explicitly. Furthermore, our participants are mainly students and not professional programmers. However, the lack of years of experience could be an advantage when we only present short programs, as it still leads to sufficiently long thinking in the scanner. Our participant sample is also in line with the majority of published [fMRI](#) studies on programmers.





## FMRI STUDY DESIGN

---

### 7.1 REQUIREMENTS FOR AN FMRI EXPERIMENT DESIGN

After conducting our prestudy to clarify whether the tasks presented here can be used as a baseline for program comprehension, we plan to conduct an **fMRI** study in the near future. For this **fMRI** study, the task selection and experiment design will be derived and presented in this chapter.

The first step is to decide, based on the data from the prestudy, which tasks can be used for the **fMRI** study. In particular, the time participants need to solve the tasks is important for us. If they are too short, we cannot measure a sufficient level of the **BOLD** signal, if they are too long, we might waste time in the scanner without generating more data points. In addition, we can only use the data from the correctly answered questions for later evaluations, so the correctness rate is very relevant to us.

In the **fMRI** study, the participants perform the tasks while lying in an **fMRI** scanner that measures their **BOLD** signal. The scanner times are time and cost intensive, so an **fMRI** study must be well prepared. Therefore, we conducted an extensive prestudy to test the tasks, and we now choose the most promising ones. We aim at participants answering as many tasks as possible correctly to maximize the collected data in each session. Since in an **fMRI** study often no more than 20 participants are invited, every data point is important. In addition, participants must be carefully screened beforehand to determine if they are suitable. We must avoid discarding data, because of criteria that could have been avoided with the right selection of participants. First of all, the participant must not be claustrophobic and must be medically suitable for an **fMRI** scan (e.g., no cardiac pacemaker). Furthermore, own private glasses cannot be worn in the scanner, therefore the lab has to provide special **fMRI** suitable glasses. A relevant factor for the study is, as for the prestudy, the minimum English level and the necessary programming experience. Besides, the participant should not be familiar with the tasks, that is they should not have taken part in the prestudy.

### 7.2 TASK SELECTION

Besides the selection criteria of the participants, we also have to take a close look at the tasks which we will present. In this section, we provide an overview about different viewpoints how long these tasks should take to solve and we explain our criteria for the tasks of our **fMRI** study.

According to the paper by Smith et al. [41], we need a task that is at least 10 seconds long to acquire a sufficiently strong **BOLD** signal. This must be followed by a rest phase of at least 10 seconds in order for the **BOLD** signal to drop to the baseline and not interfere

with the subsequent task. Amaro, and Barker [1] advises not to exceed 2 *minutes* per task, not to exceed 12 *minutes* per run, and the total time should not be longer than 40 minutes. Additionally, the paper by Siegmund et al. [39] recommends a maximum time of 30 *minutes* in the fMRI scanner, because the participants will start to get restless and the resulting motion artifacts strongly increases.

We should also consider the initialization time, which the scanner needs before we can start the tasks. This time should be around 9 *minutes*. The BOLD signal needs at least 5 *seconds* to be build up properly, and after the task again 12 *seconds* until it re-arrives at its initial position. Therefore, we should let the participant rest between the tasks for at least 12 *seconds*. Otherwise, the BOLD signal would be biased by the preceding task. The tasks must be short enough to be displayed on the screen (a special fMRI compatible screen) with a maximum of 18 lines, and the participants should not scroll as this activates other brain areas (i.e., motor cortex) and could dilute the observed brain activation [38].

The task must be answered correctly, otherwise we have no control whether the participant really did what we asked him to do or did something else. This requirement has already been taken into account in the design of the prestudy, so that the prestudy is as similar as possible to the fMRI study. The further runs are important to increase the statistical power of the experiment in order to obtain a significant result. It is suggested to aim for a length of 1 to 2 minutes for program comprehension tasks [37].

There is also evidence that shows data of the BOLD signal are not of high quality already after a task length of over one minute, because the concentration of the participants decreases. Thus, the tasks are chosen to be completable within one minute and they are also automatically terminated after one minute. If researchers split an fMRI session into multiple runs, the participants usually want to move. This is undesired in fMRI studies. As soon as the participant moves too much or leaves the device, it has to be reinitialized. Therefore, in the previous studies by Peitek [29] and Siegmund et al. [36], there was no longer a pause during the study, but a longer rest between each task, 20 to 30 instead of 10 seconds. However, the total time of the tasks including breaks was not longer than 30 minutes.

The *comprehension times* and the correctness rates of the prestudy serve as a basis for the selection of tasks for the fMRI study. The following criteria are used:

- *C1a: comprehension time  $q_1 \geq 10$  seconds*<sup>13</sup>  
The first quartiles of the *comprehension time* distribution should be above 10 *seconds*, which means that at least 75% of the participants in this task took longer than 10 *seconds*. This is important because otherwise the BOLD signal is not strong enough.
- *C2a: correctness rate  $cr \geq 75\%$*   
Since we can only use the data for which the task was answered correctly, the correctness rate is very important. It is expected that these tasks should be answered correctly by at least 75% of the participants.

<sup>13</sup>  $q_1$  is the lower quartile of the data set distribution. 25% of the data points lie below  $q_1$ , see also the visualization on a boxplot in Section 6.1.2

- *C3a: comprehension time  $q3 \leq 60$  seconds<sup>14</sup>*

The third quartiles of the *comprehension time* distribution should be below 60 *seconds*, which means the time required to complete the task is less than 60 *seconds* for 75% of the participants, otherwise the task will take too long and we will not get through as many tasks.

Criteria *C1a*, *C2a*, and *C3a* were applied to the prestudy data and yielded the following number of possible tasks. For the mathematical tasks there are 4 out of 20 and for *RPM*, there are 3 out of 20 suggested tasks. There are 8 out of 20 for program comprehension and 12 out of 20 for language-oriented tasks. These tasks are presented in the *fMRI* study sorted descending by correctness rate.

The criteria we chose are very high, so we did not select enough mathematical, program comprehension, and *RPM* tasks for our study in the mathematical and *RPM* tasks. The following are weakened criteria to select a sufficient number of tasks.

- *C1b: comprehension time  $q2 \geq 10$  seconds*

The second quartiles of the *comprehension time* distribution should be above 10 *seconds*, which means that at least 50% of the participants in this task took longer than 10 *seconds*. In *C1a* we expected that 75% of the participant took longer than 10 *seconds*.

- *C2b: correctness rate:  $cr \geq 70\%$*

It is expected that these tasks should be answered correctly by at least 70% of the participants. In criteria *C2a* we aimed at a correctness rate of 75%.

- *C3b: comprehension time  $q3 \leq 70$  seconds*

The third quartiles of the *comprehension time* distribution should be below 70 *seconds*, which means the time required to complete the task is less than 70 *seconds* for 75% of the participants, this is 10 *seconds* longer than in criteria *C3a*.

Although these criteria have now been lowered, they nevertheless still have high requirements. The probability that the task will be solved in less than 10 *seconds* is increasing, although when looking at the exact data, the times were not actually much less than 10 *seconds*, even in the first quartile. The correctness rate is lowered by 5%, but we still assume a very high correctness rate of 70%. The maximum processing time is increased by 10 *seconds*, so it can happen that more participants do not finish the task before we automatically switch to the next one. In this case, however, a message is displayed that the task is about to end, so that the participants are not surprised by the automatic switching to the next task.

With the new criteria, an additional 5 tasks for program comprehension are possible. Also 8 more tasks for *RPM* and 12 additional mathematical tasks are suitable. These tasks are also presented sorted by correctness rate in the *fMRI* study. Therefore, we have 11 out of 20 *RPM*, 13 out of 20 program comprehension tasks, 16 out of 20 mathematical tasks, and 12 out of 20 language-oriented tasks.

<sup>14</sup>  $q3$  is the upper quartile of the data set distribution. 25% of the data points lie above  $q3$ , see also the visualization on a boxplot in Section 6.1.2

Task	Mean Time (s)	q1 (s)	q2 (s)	q3 (s)	maximum (s)	correct rate (%)	C1a	C2a	C3a	C1b	C2b	C3b
Math 16	14.71	10.35	12.77	16.46	25.61	90.70%						
Math 10	16.53	10.03	13.94	18.72	31.75	84.62%						
Math 1	23.17	12.78	19.77	25.28	44.03	83.72%						
Math 13	17.52	10.50	14.83	19.20	32.25	83.72%						
Math 2	13.27	8.25	11.07	15.40	26.13	97.62%						
Math 9	14.10	8.76	11.79	16.63	28.42	97.44%						
Math 12	13.17	8.36	11.11	13.55	21.34	95.35%						
Math 6	13.41	9.03	11.97	16.71	28.23	94.87%						
Math 8	16.84	9.07	12.28	19.21	34.41	94.87%						
Math 14	12.15	8.21	10.86	12.97	20.10	90.70%						
Math 18	15.02	9.56	12.41	18.20	31.17	90.70%						
Math 17	12.01	8.49	10.56	13.07	19.93	86.05%						
Math 5	12.75	8.76	10.78	15.61	25.89	84.62%						
Math 3	14.57	9.17	12.85	17.73	30.58	82.93%						
Math 20	13.65	8.57	10.54	14.00	22.15	76.74%						
Math 11	11.37	8.77	10.52	13.02	19.40	74.42%						
Math 4	11.19	7.26	9.58	13.43	22.69	97.56%						
Math 15	10.89	7.20	8.97	12.47	20.36	95.35%						
Math 19	12.43	7.83	9.62	12.90	20.51	95.35%						
Math 7	11.15	7.77	9.26	11.97	18.27	84.62%						

Figure 7.1: The mathematical tasks sorted according to the criteria.

The left column shows the number that uniquely identifies the task. Now we see the mean processing time and the three quartiles. In the first and second quartiles all times under 10 seconds are marked red for clarity. The maximum time is the time by which outliers were detected. If the processing time is longer than the maximum, this is an outlier. Then the correctness rate follows, where we use a color scale, meaning that green is a higher correctness rate than red. The next 6 columns are the criteria, if they cannot be fulfilled, a gray bar appears. The name of the task is colored in dark green if the C1a to C3a criteria are met and colored in light green if not but the C1b to C3b criteria are fulfilled. If the task does not suit either the strong nor the relaxed criteria it is colored in grey.

The underlying data, and which tasks were accepted, is shown in a chart. For the language-oriented tasks, only the first criteria were applied because more than enough tasks were already accepted (Figure 7.2). In the program comprehension tasks (Figure 7.4) we notice that especially the upper limit of the processing time is mostly a problem. In contrast, in the RPM tasks (Figure 7.3), we find that the minimum time that a task should take is challenging. In the mathematical tasks (Figure 7.1) this is a major issue. With the looser criteria, we have achieved enough tasks for the fMRI study, but we should consider lengthening the mathematical tasks by adding and subtracting more digits in a row, currently there are seven digits. For example, it is possible to do an additional prestudy in which calculations up to 20 digits are tested. Thereby the attention should be focused on the correctness rate, which should not be forgotten although the times are longer.

Task	Mean Time (s)	q1 (s)	q2 (s)	q3 (s)	maximum (s)	correct rate (%)	C1a	C2a	C3a
Artificial Intelligence	22.32	13.29	20.90	28.04	50.15	97.67%			
Geology	18.91	13.53	17.73	22.21	35.23	97.67%			
Diary	18.51	11.41	15.79	22.88	40.10	95.35%			
Graffiti	18.01	12.17	16.53	21.44	35.35	95.35%			
Transportation	33.22	15.82	22.61	38.36	72.17	95.12%			
Planet	18.07	13.38	16.68	21.03	32.50	90.70%			
Frost	17.26	11.85	15.20	19.89	31.96	90.70%			
Protein	26.47	16.10	27.42	34.35	61.72	89.74%			
Cartoon	21.56	15.58	19.76	25.82	41.20	88.37%			
Acting	28.53	17.67	23.96	36.80	65.50	84.62%			
Planetarium	24.42	15.46	21.41	30.86	53.96	84.62%			
Blog	26.37	15.81	22.40	31.97	56.20	79.07%			
Photosynthesis	13.04	6.29	8.48	14.41	26.60	100.00%			
Sun	12.07	6.66	9.89	14.76	26.91	100.00%			
Paralympic Games	11.05	5.37	7.71	11.65	21.09	97.67%			
Microscope	15.45	9.94	14.55	18.98	32.54	97.62%			
Gravity	13.39	7.99	12.36	18.48	34.21	95.35%			
Biathlon	13.63	7.30	10.34	15.59	28.03	95.35%			
Windmill	14.25	7.44	12.70	19.71	38.11	71.79%			
Security System	29.19	16.99	21.93	33.95	59.40	26.83%			

Figure 7.2: The language-oriented tasks sorted according to the criteria.

Task	Mean Time (s)	q1 (s)	q2 (s)	q3 (s)	maximum (s)	correct rate (%)	C1a	C2a	C3a	C1b	C2b	C3b
Medium 3	18.62	11.12	15.90	21.51	37.10	93.02%						
Hard 2	25.75	13.43	24.26	31.13	57.68	84.62%						
Hard 3	39.51	19.28	33.77	54.76	107.99	81.40%						
Easy 1	14.15	7.01	12.62	20.46	40.65	95.35%						
Medium 4	14.87	8.28	11.65	19.42	36.13	95.35%						
Medium 1	21.95	9.22	14.15	25.37	49.59	92.68%						
Easy 2	12.06	6.81	10.50	14.29	25.49	90.24%						
Medium 5	13.35	7.09	11.07	16.12	29.67	81.40%						
Easy 5	16.32	9.54	13.29	19.03	33.27	79.07%						
Special 3	24.85	14.11	22.24	29.79	53.31	74.42%						
Hard 4	18.83	12.97	18.09	22.73	37.37	74.42%						
Easy 4	9.86	6.74	8.42	11.68	19.08	100.00%						
Easy 7	7.06	3.90	5.25	9.27	17.33	97.67%						
Easy 6	9.57	5.48	7.23	12.48	22.97	88.37%						
Easy 3	11.86	7.23	9.84	14.96	26.57	76.92%						
Special 4	38.03	18.29	29.78	44.79	84.55	58.14%						
Special 1	34.75	17.37	28.46	41.83	78.52	56.41%						
Medium 2	43.84	10.48	19.57	34.23	69.84	51.28%						
Special 2	59.83	29.09	45.45	86.50	172.62	56.41%						
Hard 1	68.60	29.69	46.10	90.43	181.54	40.48%						

Figure 7.3: The RPM tasks sorted according to the criteria. In the third quartiles the times over 60 seconds are marked red.

Task	Mean Time (s)	q1 (s)	q2 (s)	q3 (s)	maximum (s)	correct rate (%)	C1a	C2a	C3a	C1b	C2b	C3b
CountVowels	32.70	16.66	29.25	44.56	86.40	95.12%						
Palindrome	43.53	29.59	39.91	54.81	92.63	92.31%						
Factorial	19.17	11.19	16.89	22.99	40.69	90.70%						
Hurricane	35.16	24.84	31.93	45.18	75.69	84.62%						
YesNo	29.65	21.16	29.03	36.54	59.62	84.62%						
Fibonacci	36.99	19.70	34.14	42.46	76.59	81.40%						
CrossSum	40.16	18.34	32.54	50.51	98.76	79.49%						
Power	42.90	25.20	34.38	48.40	83.19	79.07%						
ArrayAverage	48.41	30.12	43.89	63.93	114.65	83.72%						
SquareRoot	50.07	32.40	42.33	61.79	105.88	79.07%						
Prime	52.14	30.36	45.01	65.64	118.55	79.07%						
BinaryToDecimal	57.40	33.67	50.18	63.75	108.88	79.07%						
ForwardBackward	50.26	33.19	40.36	61.28	103.41	72.09%						
DumbSort	72.86	40.76	66.66	93.17	171.78	80.49%						
EverySecondChar	52.94	28.53	36.41	50.62	83.76	60.47%						
AlwaysZero	38.65	25.02	33.78	42.44	68.57	58.97%						
LCM	61.31	36.02	48.34	68.18	116.43	65.12%						
ContainSubstring	118.63	56.81	87.73	170.31	340.57	69.05%						
Fast	93.19	61.53	78.81	116.37	198.63	65.12%						
CountLastWord	91.27	58.78	80.38	114.56	198.23	48.72%						

Figure 7.4: The program comprehension tasks sorted according to the criteria.

### 7.3 DERIVED FMRI STUDY DESIGN

The **fMRI** study is designed in a block design, as well as the prestudy. This means that the four tasks alternate. This way it will not be so monotonous for the participant. Furthermore the activation in the brain drifts <sup>15</sup> as the experiment goes on, so by alternating the tasks all the tasks will suffer almost equally from this phenomenon. If we would, for example, start first with all mathematical tasks, then all language-oriented tasks followed by all the **RPM** task and finally all the program comprehension tasks, the data of the program comprehension tasks will be unequally affected by this phenomenon than the mathematical tasks. Therefore, we must alternate the tasks to compare them against each other. A few features apply to all tasks. To ensure that the **fMRI** data is of high quality, we filter out data where the subject did not understand or certainly did not complete the task. Therefore, the procedure of each task is the same. The task starts and the task instruction is displayed. With one click the participant can continue, but after a period of time determined by the prestudy, the task is automatically switched on. After that the question with 4 answer options is shown, the three further answer options from the prestudy are omitted. If the participant does not know what they should not click. Thus, we save ourselves a button and thus complicated button press sequences. It is automatically switched on after max time, but after one minute at the latest. There is a 20 *seconds* rest before the next task starts.

In this schedule there are 9 blocks. The tasks were sorted as explained above, by meeting the criteria and by correctness rate. In this order they were included in the derived experiment design (Figure 7.5). With a mean *comprehension* and *answer time*, the participant should need 28.5 *minutes* including breaks for all the tasks. If the theoretical maximum time is required everywhere, whereby the individual task is switched on after 1 *minute* at the latest, we arrive at 42.3 *minutes* for all the tasks. Overall, during the blocks, the participants

<sup>15</sup> as illustrated on <http://brainvoyager.com/bv/doc/UsersGuide/Preprocessing/TemporalHighPassFiltering.html>, last visited: 03.03.2021

Block no.	1	2	3	4	5	6	7	8	9
<b>Math-oriented</b>	Math 16	Math 10	Math 1	Math 13	Math 2	Math 9	Math 12	Math 6	Math 8
Rest	20 seconds								
<b>Language-oriented</b>	Artificial Intelligence	Geology	Diary	Graffiti	Transportation	Planet	Frost	Protein	Cartoon
Rest	20 seconds								
<b>Program Comprehension</b>	CountVowels	Palindrome	Factorial	Hurricane	YesNo	Fibonacci	CrossSum	Power	ArrayAverage
Rest	20 seconds								
<b>RPM</b>	Medium 3	Hard 2	Hard 3	Easy 1	Medium 4	Medium 1	Easy 2	Medium 5	Easy 5
Rest	20 seconds								
<b>Sum of mean times (minutes)</b>	3.00	3.31	3.27	3.00	3.03	3.28	3.01	3.19	3.38
<b>Sum of max. time (minutes)</b>	4.51	4.80	4.96	4.53	4.62	5.30	4.14	4.76	4.73

Figure 7.5: The resulting time table with time annotations.

The sum of mean time is calculated by adding the four tasks mean times plus the four times 20 seconds rest time. The sum of max time was calculated analogously, whereby tasks with a max time longer than 1 *minute* were shortened to 1 *minute*.

will spend 13.3 *minutes* resting between tasks. Which leaves 15.1 to 29 *minutes* for solving tasks. To summarize, we can assume that the participants will take about the same time as in the mean calculation, but very unlikely longer than in the max calculation.

#### 7.4 FMRI STUDY ONSITE

The onsite procedure consists of more than just the presented *fMRI* session. The participant will fill out the consent form and demographics prior to the *fMRI* experiment. Next, a sample task is presented on a PC so that possible questions about the general design of the tasks can be clarified. If everything is clear, the participant enters the scanner. First of all, there is an initialization, we calculate a generous 10 *minutes* for this. We have no influence on this time as it is a necessary anatomical scan to measure each participant's individual brain structure. Then the actual study with the experiment design described in the previous section begins. After 42.3 *minutes* at the latest, the study is completed. Resulting in an overall maximum time of 52.3 *minutes* inside the *fMRI* scanner. According to Amaro, and Barker [1], this is the maximum time a person should solve tasks in the scanner. If the study lasts too long, the concentration fades and the data quality is reduced. Afterwards a questionnaire on the PC or verbally by the study leader follows, in which the personal experience in the scanner is assessed, as well as, if it was possible to solve the tasks without having *task-unrelated thoughts* (TUT) [4]. Furthermore, the participant might want to share their ideas on what we are testing. This feedback from the participant can help us to constantly improve our design in follow-up studies.

In this [fMRI](#) study we will look at the areas that are activated during each task type. Since we do not have a baseline yet, we need to evaluate them against rest, in our case cross-fixation. We then normalize them across all tasks of the same type and across all participants. Therefore, we can first try to reproduce the results of the literature review. Afterwards, we try to construct the activation during program comprehension from the activation during the three other task types using *Elastic Net*. If this works, this reconstruction of program comprehension activation will be our new baseline, which can be used to conduct experiments that investigate differences such as differences in the comprehension of iterative vs. recursive implemented functions or the influence of beacons. The advantage would be that the general activation is reduced by subtracting the baseline and the small differences are more visible.



## CONCLUDING REMARKS

---

### 8.1 CONCLUSION

In this thesis, we highlighted the importance of choosing the right baseline in *fMRI* studies to obtain reliable and comparable results. With regard to *fMRI* studies investigating program comprehension, we presented 3 tasks from the domains of mathematical, reading and mental load. These areas should, based on our literature review of related research, activate similar areas as in program comprehension and thus provide a good baseline.

Afterwards, we highlighted the areas of mathematics, language and mental load in our literature review. Then we created 20 tasks for each of them, with the aim of using them as a baseline in future *fMRI* studies. Therefore, we first need to test them in a *fMRI* study. Since the *fMRI* studies are very time-consuming, we set up an online study with which we test these tasks in advance.

In this study, 20 tasks of each type were presented. The answers and response times of the 351 participants were thereby recorded. In the evaluation, the data first went through a validation pipeline. In this way, all unreliable data sets were discarded, resulting in 120 data sets for the evaluation. Afterwards, the correctness rate and the *comprehension time* were calculated for each task. We obtained the result of a consistently high correctness rate. The *comprehension times* were generally within the limits recommended for an *fMRI* study. On the basis of the correctness rate and *comprehension time*, we designed an *fMRI* study.

The design of the *fMRI* study follows the block design of the online study. The nine most appropriate tasks of each type were entered into a study schedule. With the help of the *comprehension times*, an average and a maximum time analysis was performed for the *fMRI* study to maximize the use of the recommended times in the scanner without exceeding them.

As a side note, we also used the results of the online study for further analysis. We predict the performance of the program comprehension tasks using *Elastic Net*. This worked tolerably well based on the performance in the mathematical task, language-oriented task and mental load task. Thereby mental load has a large influence. For a correlation with the demographic data, we probably have too few data points to train the algorithm properly.

### 8.2 FUTURE WORK

Mainly, this *fMRI* study can be conducted to test the hypothesis that similar areas of the brain are activated in program comprehension and when doing mathematics, reading and *RPM*. Thus, in the later process, brain activation changes in the program comprehension

task are better detectable. If this is the case, these tasks can serve as a baseline to investigate the influence of beacons, syntax highlighting or other features on the cognitive load during program comprehension.

Another aspect that can be investigated is the possible different activation during *comprehension time* and *answer time*. Are there different activations, how do they differ, and to what extent are they similar for different types of tasks?

APPENDIX

---

The Appendix contains

- All used tasks
- Screenshots of the online survey of the prestudy
- Detailed Listings of the demographics
- Figures showing the *comprehension time* and the *answer time*
- Figures showing the selected answers
- The *Jupyter Notebook* that was used for the evaluation

Since these are also large documents, they can be accessed via this link:  
<https://ma-apdx-ab.bergum.de>



## BIBLIOGRAPHY

---

- [1] Edson Amaro and Gareth Barker. "Study Design in fMRI: Basic Principles." In: *Brain and cognition* 60 (2006), pp. 220–232.
- [2] Alfredo Ardila, Byron Bernal, and Monica Rosselli. "How Localized Are Language Brain Areas? A Review of Brodmann Areas Involvement in Oral Language." In: *Archives of Clinical Neuropsychology* 31 (2016), pp. 112–122.
- [3] Alan Baddeley. "The Episodic Buffer: A New Component of Working Memory?" In: *Trends in cognitive sciences* 4 (2000), pp. 417–423.
- [4] Jeffrey Binder, Julia Frost, Thomas Hammeke, PSF Bellgowan, Stephen Rao, and Robert Cox. "Conceptual Processing During the Conscious Resting State: A Functional MRI Study." In: *Journal of cognitive neuroscience* 11 (1999), pp. 80–93.
- [5] Corinna Bonhage, Jutta Mueller, Angela Friederici, and Christian Fiebach. "Combined Eye Tracking and fMRI reveals Neural Basis of Linguistic Predictions During Sentence Comprehension." In: *Cortex* 68 (2015), pp. 33–47.
- [6] Joao Castelhana, Isabel Duarte, Carlos Ferreira, Joao Duraes, Henrique Madeira, and Miguel Castelo-Branco. "The Role of the Insula in Intuitive Expert Bug Detection in Computer Code: An fMRI Study." In: *Brain imaging and behavior* 13 (2019), pp. 623–637.
- [7] Mark Cohen, Stephen Kosslyn, Hans Breiter, Gregory DiGirolamo, William Thompson, AK Anderson, SY Bookheimer, Bruce Rosen, and JW Belliveau. "Changes in Cortical Activity During Mental Rotation A Mapping Study Using functional MRI." In: *Brain* 119 (1996), pp. 89–100.
- [8] Jody Culham, Patrick Cavanagh, and Nancy Kanwisher. "Attention Response Functions: Characterizing Brain Areas Using fMRI Activation During Parametric Variations of Attentional Load." In: *Neuron* 32 (2001), pp. 737–745.
- [9] Nicole Davis, Christopher Cannistraci, Baxter Rogers, Christopher Gatenby, Lynn Fuchs, Adam Anderson, and John Gore. "The Neural Correlates of Calculation Ability in Children: An fMRI Study." In: *Magnetic resonance imaging* 27.9 (2009), pp. 1187–1197.
- [10] Benjamin Floyd, Tyler Santander, and Westley Weimer. "Decoding the Representation of Code in the Brain: An fMRI Study of Code Review and Expertise." In: *Proceedings of the 39th International Conference on Software Engineering, ICSE 2017, Buenos Aires, Argentina, May 20-28, 2017*. IEEE / ACM, 2017, pp. 175–186.
- [11] Jean Gibbons and S. Chakraborti. "Comparisons of the Mann-Whitney, Student's t, and Alternate t Tests for Means of Normal Distributions." In: *The Journal of Experimental Education* 59 (1991), pp. 258–267.
- [12] Peter Hagoort. "Nodes and Networks in the Neural Architecture for Language: Broca's Region and Beyond." In: *Current opinion in Neurobiology* 28 (2014), pp. 136–141.
- [13] Peter Hagoort and Peter Indefrey. "The Neurobiology of Language Beyond Single Words." In: *Annual review of neuroscience* 37 (2014), pp. 347–362.

- [14] Yu Huang, Xinyu Liu, Ryan Krueger, Tyler Santander, Xiaosu Hu, Kevin Leach, and Westley Weimer. "Distilling Neural Representations of Data Structure Manipulation Using fMRI and fNIRS." In: *Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019*. IEEE / ACM, 2019, pp. 396–407.
- [15] Yoshiharu Ikutani, Takatomi Kubo, Satoshi Nishida, Hideaki Hata, Kenichi Matsumoto, Kazushi Ikeda, and Shinji Nishimoto. "Expert Programmers Have Fine-Tuned Cortical Representations of Source Code." In: *bioRxiv* (2020).
- [16] Kimberle M. Jacobs. "Brodmann's Areas of the Cortex." In: *Encyclopedia of Clinical Neuropsychology*. Ed. by Jeffrey Kreutzer, John DeLuca, and Bruce Caplan. Springer New York, 2011, pp. 459–459.
- [17] Stephen Kosslyn, Alvaro Pascual-Leone, Olivier Felician, Susana Camposano, Julian Keenan, G Ganis, KE Sukel, NM Alpert, et al. "The Role of Area 17 in Visual Imagery: Convergent Evidence from PET and rTMS." In: *Science* 284 (1999), pp. 167–170.
- [18] James Kroger, Leigh Nystrom, Jonathan Cohen, and Philip Johnson-Laird. "Distinct Neural Substrates for Deductive and Mathematical Processing." In: *Brain research* 1243 (2008), pp. 86–103.
- [19] Frank Krueger, Vittoria Spampinato, Matteo Pardini, Sinisa Pajevic, Jacqueline Wood, George Weiss, Steffen Landgraf, and Jordan Grafman. "Integral Calculus Problem Solving: An fMRI Investigation." In: *Neuroreport* 19 (2008), p. 1095.
- [20] Ryan Krueger, Yu Huang, Xinyu Liu, Tyler Santander, Westley Weimer, and Kevin Leach. "Neurological Divide: An fMRI Study of Prose and Code Writing." In: *ICSE '20: 42nd International Conference on Software Engineering, Seoul, South Korea, 27 June - 19 July, 2020*. ACM, 2020, pp. 678–690.
- [21] Yun-Fei Liu, Judy Kim, Colin Wilson, and Marina Bedny. "Computer Code Comprehension Shares Neural Resources with Formal Logical Inference in the Fronto-Parietal Network." In: *BioRxiv* (2020).
- [22] Alex Martin. "Automatic Activation of the Medial Temporal Lobe During Encoding: Lateralized Influences of Meaning and Novelty." In: *Hippocampus* 9 (1999), pp. 62–70.
- [23] Alex Martin, Cheri Wiggs, and Jill Weisberg. "Modulation of Human Medial Temporal Lobe Activity by Form, Meaning, and Experience." In: *Hippocampus* 7 (1997), pp. 587–593.
- [24] Laura Matzen, Zachary Benz, Kevin Dixon, Jamie Posey, James Kroger, and Ann Speed. "Recreating Raven's: Software for Systematically Generating Large Numbers of Raven-like Matrix Problems with Normed Properties." In: *Behavior research methods* 42 (2010), pp. 525–541.
- [25] Aaron Newman, Roumyana Pancheva, Kaori Ozawa, Helen Neville, and Michael Ullman. "An Event-Related fMRI Study of Syntactic and Semantic Violations." In: *Journal of psycholinguistic research* 30 (2001), pp. 339–364.
- [26] Sharlene Newman, Gregory Willoughby, and Benjamin Pruce. "The effect of problem structure on problem-solving: an fMRI study of word versus number problems." In: *Brain research* 1410 (2011), pp. 77–88.

- [27] Chris Parnin, André Brechmann, Norman Peitek, Sven Apel, and Janet Siegmund. "Program Comprehension and Code Complexity Metrics: An fMRI Study." In: *Proceedings of the International Conference on Software Engineering (ICSE)*. IEEE, May 2021. Acceptance rate: 23 Prozent (138 of 602); to appear. 2021.
- [28] Christoph Pabst. "Magnetresonanz-Tomographie." In: *Lernskript für Mediziner: Grundlagen der Magnetresonanz-Tomographie* (2013).
- [29] Norman Peitek. "A Neuro-Cognitive Perspective of Program Comprehension." In: *Proceedings of the 40th International Conference on Software Engineering: Companion Proceedings, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018*. ACM, 2018, pp. 496–499.
- [30] Norman Peitek, Janet Siegmund, Sven Apel, Christian Kästner, Chris Parnin, Anja Bethmann, Thomas Leich, Gunter Saake, and André Brechmann. "A Look into Programmers' Heads." In: *IEEE Trans. Software Eng.* 46 (2020), pp. 442–462.
- [31] Vivek Prabhakaran, Jennifer Smith, John Desmond, Gary Glover, and John Gabrieli. "Neural Substrates of Fluid Reasoning: An fMRI Study of Neocortical Activation During Performance of the Raven's Progressive Matrices Test." In: *Cognitive psychology* 33 (1997), pp. 43–63.
- [32] Cathy Price. "A Review and Synthesis of the First 20 Years of PET and fMRI Studies of Heard Speech, Spoken Language and Reading." In: *NeuroImage* 62 (2012), pp. 816–847.
- [33] John Raven. "The Raven's Progressive Matrices: Change and Stability over Culture and Time." In: *Cognitive psychology* 41 (2000), pp. 1–48.
- [34] TC Rickard, SG Romero, G Basso, C Wharton, S Flitman, and J Grafman. "The Calculating Brain: An fMRI Study." In: *Neuropsychologia* 38 (2000), pp. 325–335.
- [35] Janet Siegmund. "Framework for Measuring Program Comprehension." PhD thesis. Otto von Guericke University Magdeburg, 2012.
- [36] Janet Siegmund, Sven Apel, Christian Kästner, Chris Parnin, Anja Bethmann, Gunter Saake, Thomas Leich, and André Brechmann. "Measuring Program Comprehension with Functional Magnetic Resonance Imaging." In: *Software Engineering & Management 2015, Multikonferenz der GI-Fachbereiche Softwaretechnik (SWT) und Wirtschaftsinformatik (WI), FA WI-MAW, 17. März - 20. März 2015, Dresden, Germany*. Vol. P-239. GI, 2015, pp. 63–64.
- [37] Janet Siegmund, André Brechmann, Sven Apel, Christian Kästner, Jörg Liebig, Thomas Leich, and Gunter Saake. "Toward Measuring Program Comprehension with Functional Magnetic Resonance Imaging." In: *20th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-20), SIGSOFT/FSE'12, Cary, NC, USA - November 11 - 16, 2012*. ACM, 2012, p. 24.
- [38] Janet Siegmund, Christian Kästner, Sven Apel, Chris Parnin, Anja Bethmann, Thomas Leich, Gunter Saake, and André Brechmann. "Understanding Understanding Source Code with Functional Magnetic Resonance Imaging." In: *36th International Conference on Software Engineering, ICSE '14, Hyderabad, India - May 31 - June 07, 2014*. ACM, 2014, pp. 378–389.

- [39] Janet Siegmund, Norman Peitek, Chris Parnin, Sven Apel, Johannes Hofmeister, Christian Kästner, Andrew Begel, Anja Bethmann, and André Brechmann. “Measuring Neural Efficiency of Program Comprehension.” In: *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2017, Paderborn, Germany, September 4-8, 2017*. ACM, 2017, pp. 140–150.
- [40] Janet Siegmund, Norbert Siegmund, and Sven Apel. “How Reviewers Think About Internal and External Validity in Empirical Software Engineering.” In: *Software Engineering 2016, Fachtagung des GI-Fachbereichs Softwaretechnik, 23.-26. Februar 2016, Wien, Österreich*. GI, 2016, pp. 83–84.
- [41] Stephen Smith, Mark Jenkinson, Christian Beckmann, Karla Miller, and Mark Woolrich. “Meaningful design and contrast estimability in fMRI.” In: *NeuroImage* 34.1 (2007), pp. 127–136.
- [42] Craig Stark and Larry Squire. “When Zero Is Not Zero: The Problem of Ambiguous Baseline Conditions in fMRI.” In: *Proceedings of the national Academy of Sciences* 98 (2001), pp. 12760–12766.
- [43] John Sweller. “Cognitive Load Theory, Learning Difficulty, and Instructional Design.” In: *Learning and instruction* 4 (1994), pp. 295–312.
- [44] Robert Whelan. “Neuroimaging of Cognitive Load in Instructional Multimedia.” In: *Educational Research Review* 2 (2007), pp. 1–12.
- [45] Laure Zago, Mauro Pesenti, Emmanuel Mellet, Fabrice Crivello, Bernard Mazoyer, and Nathalie Tzourio-Mazoyer. “Neural Correlates of Simple and Complex Mental Calculation.” In: *Neuroimage* 13 (2001), pp. 314–327.
- [46] Karl Zilles. “Brodmann: A Pioneer of Human Brain Mapping—His Impact on Concepts of Cortical Organization.” In: *Brain* 141 (2018), p. 3262.
- [47] Hui Zou and Trevor Hastie. “Regularization and Variable Selection via the Elastic Net.” In: *Journal of the royal statistical society: series B (statistical methodology)* 67 (2005), pp. 301–320.