Universiteit Leiden
The Netherlands

Opleiding Informatica

An exploratory research on Reading Code

Matthijs de Keijzer

Supervisor:
Felienne Hermans

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

23/06/2022

1 PREFACE

This thesis will go in further depth on how students read code. I have always been intrigued by how students learn to code. I have given a couple of workshops about learning Python code, so I found it interesting to use what I learned in this thesis and apply it there.

I would like to thank my supervisors for their guidance and support during this process. I also wish to thank the students that took the time to take the interview. And I also would like to thank B. Heemskerk for his cooperation in enabling me to have the interviews at a high school.

2 ABSTRACT

Programming is becoming increasingly important. Reading code is an important skill to learn programming. Reading code improves other programming skills. But not much research has been done on how high school students read code. This research explored how high school students read and comprehend code. To do this multiple talk-aloud interviews with high school students have been held. The students were given three different types of code that each tackled different reading problems. After this, the interviews were analyzed and broken down into multiple larger categories. This study confirmed that most novice programmers read code in a similar way as reading natural language (top to bottom). This was mainly true whilst reading through the code for the first time. After this first time, the reading order sometimes deviated from this and followed the execution flow. Illogical code was not followed by the students, although it may have resulted in some indirect mistakes. Almost all students made a note on the illogical code. Most functions and methods that the students did not understand were guessed. It depended mainly on the name of the function or method, but also the name of the variable, whether an attempt was made to guess the meaning. Sometimes unnecessary pieces of code were used to complete the code. This despite sometimes introducing errors in the rest of the code. Explanations were given as to why these unnecessary pieces were used, but a concrete conclusion cannot be taken. Code was skipped if the code had too many variables or parts that needed to be stored in memory. This even though the skipped code was essential to successfully completing the code. It could mean that exactly following or understanding code with many functions, loops, and variables is a very difficult or taxing problem. This research could be improved by, among other things, interviewing more subjects and subjects of different high schools.

3 CONTENTS

4 INTRODUCTION

Programming and software are becoming increasingly important in today's society. Thus, more efforts are made to teach people how to code and how to understand software. Learning code can be done in several ways. One example is massive open online courses (MOOCs) that you can join to learn coding or other skills (Udemy, 2021). But computer science is also increasingly incorporated into the teaching curriculum. Since the introduction of the "Tweede Fase" (1998), computer science has been an optional course in upper HAVO and VWO in the Netherland (SLO, 2014). HAVO and VWO are different levels in the Dutch school system. Since 2007, it has also been a subject of choice for different learning profiles of Dutch high schools (Overheid, 2014). Therefore, some high schools have the option for high school students to be able to learn about the field of computer science and in particular learn how to code.

There are currently no real standards on how computer science needs to be taught in high schools in the Netherlands. In SLO (2016) **i**s described what students are required to know after their graduation. This guideline leaves room for interpretation and there are multiple themes that the teacher can choose. But if other courses are anything to go by, code reading is rarely an important part of the course.

Busjahn and Schulte (2013) describe that code reading is an important part of learning code and there are many aspects of learning to program where code reading is needed. It describes that code reading is connected to comprehending programs and algorithms, or algorithmic ideas. Lister et al. (2009) further demonstrate the relationship between code reading and code explaining. Lister et al. (2014) demonstrate that some students who manifest weakness in problem-solving do so because of reading-related factors.

This paper will focus on interviews using the think-aloud method (Someren et al., 1994). This means that while reading the code, the students read aloud what they are doing. Furthermore, the interviewer can ask the interviewee question related to the code during or after the interview process. This will possibly give more reasoning to the observed code reading patterns and can result in a rationale on why the student arrived at that solution.

The interviews will be done with high school students because this is the starting phase of the learning process when it comes to programming in the Netherlands. During this phase, it is important to start learning how to read code, because that is an important method for learning programming vocabulary and learning how they are applied in the required context.

This thesis aims to try to understand how high school students read Python code. The motivation is to better understand what students struggle with during the code reading process. The following main research question will be addressed in this paper, together with sub-questions that are relevant to this main research question.

RQ1. *How do high school students read and comprehend Python code?*

a) In what order do students read and comprehend code and natural language?
b) How do students read illogical code (where illogical code contradicts common knowledge)?
c) How do students read code whose functions and methods they have never seen before?
d) How do students deal with unnecessary pieces of code?
e) How do students deal with complex code (code with a lot of variables)?

4.1 Thesis overview

This chapter contains the introduction, motivation, and general overview of the paper. This is followed by Section 6: Background and related work which discusses the related works. These related works include already done research in the field of reading in general, text comprehension, and code comprehension. Section 7: Method will give the research method and the reasoning behind this research method. This will be followed by section 8: Results. This section will contain the results of the interviews given in a table overview. Followed by other notable data collected during the interviews. And any additional data that is not listed in the tables. This thesis is concluded by section 9: Discussion which will contain a conclusion, discussion, limitations, and possible further research.

5 BACKGROUND AND RELATED WORK

5.1 Learning natural languages and reading them

Before looking at how people read code, it is first important to look at how people read natural languages. Young children only remember certain aspects of a word according to Ehri (2005). This later evolves into connecting graphemes and phonemes, this happens before or during primary school. The paper describes that most words are stored using connections and using graphemes and phonemes. Where the graphemes are capital letter groups, and the phonemes are phonetic symbols in pronunciation. This means that the spelling of a word is stored together with the pronunciation of the word. This means that the spelling of a word is stored together with the pronunciation of the word. Below is an example of the words in their graphemes and their connected phonemes.

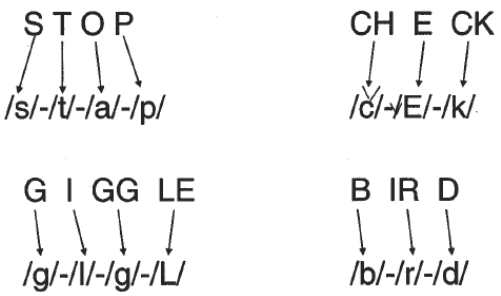


Figure 1 -Phonemes and Graphemes - Source Ehri (2005)

If this connection is made the word can get stored in memory. When these words are stored in memory, they become sight words. Sight words are words that a person sees and immediately can decode. This is because these sight words are stored in memory. The paper Ehri (2005) says that this process of decoding sight words is automatic and if the reader knows a certain word, this process cannot be stopped. If the reader knows the meaning of the words (sight words) they will not be strategic, because they can immediately decode this word. Words that the reader does not know however are not immediately able to be decoded because no sight word is stored in memory. Thus, the reader will be more strategic whilst reading the text. They will try to decode, analogize or predict the meaning of the words. Thus, to improve the speed and efficiency of reading building a large sight vocabulary is important. Muter & Snowling (1998) says that phoneme awareness and speech rate are important factors of reading skill at age of 9. Phoneme awareness is the ability to hear and manipulate the sounds in spoken words and the understanding that spoken words and syllables are made up of sequences of speech sounds (Muter & Snowling, 1998) and speech rate is the rate at which you speak. This could also translate to the reading ability of high school students; they will have a much broader sight word vocabulary. But the use of sight words and analyzing the text will still hold. In short, sight words are always preferred, but if this is not a known word then the reading will be more strategic in reading the text. The data of Baddeley (1982) suggest that phonological coding in reading was sometimes used but was in fact optional, this research was done with adult readers. Another research (Peng et al., 2018) suggests that there is a strong correlation between verbal working memory and comprehension of the text. This would suggest that knowing the pronunciation of the word will lead to a better understanding of the word and in extension the text.

5.2 Comprehending code – confusions and working memory

Hermans (2021) wrote a book on code reading and code comprehension. This book is divided into multiple chapters and four different parts. These parts are: “on reading code better”, “on thinking about code”, “on writing better code” and “on collaborating on code”. This thesis will mainly focus on the first two chapters. The first chapter “on reading code better” describes the different types of confusion and the different times of memory. These confusions are hindrances when trying to comprehend reading code the following confusions are described:

- Lack of knowledge
- Lack of information
- Lack of processing power

The lack of knowledge describes when the reader does not understand the syntax or meaning of the word. The lack of information means that the user knows what the individual parts do, but some information is obscured. This obscured information may result in confusion while reading the code. The lack of processing power means that there is not enough space in the working memory of the reader to fully

comprehend the meaning of the text. This means that there are too many variables to remember or too many functions to keep in mind.

Even though this suggests that working memory has a good influence on reading, there is still a debate about whether this is true. Despite the theoretically strong relation between reading and WM, substantial differences have been reported across studies in the size of this relation. Some studies have found that the contribution of WM to reading is very small (e.g., O'Shaughnessy & Lee Swanson, 2000), whereas other studies have found that WM almost fully explained reading performance (e.g., Daneman, 1991).

In this research, we will likely come across these confusions whilst the students are reading the code. The students are not experts in programming so they will likely have some lack of knowledge. The lack of information and the lack of processing power can be created or removed by changing the piece of programming code that the students must try to understand.

These confusions are related to different cognitive processes. Namely, a lack of knowledge means that not enough relevant facts are in the long-term memory. A lack of information is described as a challenge for the short-term memory. Information is stored in this short-term memory, but if there is too much information, some will be lost. Lastly, the book describes the working memory, this is the place where the thinking happens. This corresponds to the lack of processing power. In other papers working memory is defined as the cognitive system that holds and manipulates information (Diamond, 2013). Short-term memory is different, this is defined in Atkinson and Shiffrin (1986) as the aspect of the total memory model that holds small amounts of information to be immediately used. So, the main difference between working memory and short-term memory is that in the working memory the data can be manipulated. The size of the working memory is not very big and is limited in its capacity (Sweller et al. 1998). The book: The Programmers Brain (Hermans, 2021) claims that there can be around three to four items in this working memory.

Ayres and Paas (2009) conclude that working memory is not always the same size. It is explained that they are different between experts and novices. Information can be stored in long-term memory and can be used to expand the capacity of working memory for that domain. This is further confirmed by other research (Pollock et al., 2002, Tuovinen & Sweller, 1999). They found consistently larger working memory loads for more expert learners. So, we can conclude that working memory is improved when a person learns or knows more of a particular domain.

In general, it is not possible to increase the size or improve the effectiveness of the working memory, this is explored by Burns et al. (2017). They say that working memory-specific training led to a small effect for reading.

To conclude, working memory plays a role to solve problems that require more manipulation during the reading task. The parts of the code that are not required to be transformed are stored in the short-term

memory. But without a solid understanding, it is not possible to know how to process the text. Thus, the required context needs to have been stored in the long-term memory.

Natural language reading vs code reading

Busjahn et al. (2015) show that there is a difference between code reading and natural language reading. They found that expert programmers read the code less linearly than novice programmers. This would mean that different reading strategies are important when comparing natural language reading to the reading of code. A study was done with novice programmers with less than three years of programming experience Wu et al. (2019) concludes that to get a general idea of the code, most of the participants first used a skimming strategy. After this, the most mentioned strategy was to follow the code's execution. This can be different from reading natural language. Natural language can almost always be read from top to bottom whilst the execution flow of the code differs from the top to bottom flow. Blascheck and Sharif (2019) also tell us that compared to natural text, code readers focus more on functionality and skip unimportant constructs such as brackets. This focus on important parts of the code is further confirmed by Wu et al. (2021). This is different compared to natural language reading where the time looked at each part is distributed more evenly.

Novice code reading

To conclude, working memory plays a role to solve problems that require more manipulation during the reading task. The parts of the code that are not required to be transformed are stored in the short-term memory. But without a solid understanding, it is not possible to know how to process the text. Thus, the required context needs to have been stored in the long-term memory. This could mean that more expert programmers can keep more structures in their short-term and long-term memory. This will result in more space in the working memory and thus they will be more efficient and faster during the reading task.

Different reading strategies can be applied to better and faster comprehend the code structure. That is why the reading strategies of expert and novice programmers differ. Busjahn et al. (2015) conclude that expert programmers will read the code following the execution flow and that novice programmers will follow a more linear path through the code. This is further explored and confirmed by (Wu et al., 2021). Apart from reading strategies, (Ayers & Paas, 2009) concludes that working memory is not always the same size. It is explained that they are different between experts and novices. Information can be stored in long-term memory and can be used to expand the capacity of working memory for that domain.

Reading code is currently studied using eye trackers (Antropova et al., 2013, Busjahn et al., 2011, Guéhéneuc, 2006, Turner et al., 2014, Konandur et al., 2020, Bednarik & Tukiainen, 2006). The eye movements and eye fixations are then used to get a better understanding of the reading order and reading focus. Free University Berlin et al. (2015) used this process for novice programmers in primary school and this eye tracking is also applied with more expert programmers (Antropova et al., 2013).

Another research (Begel, 2015) says that people can recognize signs or beacons for a coding schema, which is already stored in long-term memory. This can replace several independent stored items with a single reference. And thus reduce the load on short-term memory. They also acknowledge that people read code in many different orders, including text order (like a book), control flow order (as the program executes), and data flow order (following some data values forwards or backward through the program, while ignoring others).

6 METHOD

As explained in the introduction, this research aims to find how high school students read and comprehend Python code. To answer this research question, the students will read 3 different pieces of code. The data is collected using semi-structured think-aloud interviews.

6.1 Participants

The participants that take part in this research will be 5th and 6th-year high school students. These students will be sorted by the educational level that they are in (HAVO, VWO). The students are all between 15 and 19 years old. These students all volunteered to participate in this study. The participants are required to have basic knowledge of programming and the Python programming language. This requirement is checked with the corresponding computer science teacher. All participants are Dutch students, this means that the text used in the code is in Dutch.

Students were asked during class if they wanted to participate in this study. This was done using a short presentation during the lesson of the corresponding computer science teacher. This included a brief introduction of the purpose of this research and an explanation of the bachelor thesis and computer science study.

A total of 13 interviews were conducted. All students had had 7 to 8 Python coding lessons and 4 lessons on creating their project using Python. The research was done at one high school and under the supervision of one computer science teacher. Three different fifth-year VWO classes had been asked to participate in this study. This meant that all the students had learned almost the same material. Of course, there may be variation here between different classes and some variation in the understanding of the material. The students

6.2 Code

The programming language used is Python. This programming language is chosen based on the programming language that the participants are familiar with. Although the research principles from this research could be replicated using different programming languages. Important parts of the Python programming languages are included in this program, this included: variables, if statements, while-loops, for-loops, functions, and build-in functions. The 3 different pieces of code are based on the different

confusions of The Programmers Brain (Hermans, 2021). These are lack of information, lack of processing power, and lack of knowledge.

The pieces of code were created so that the students will are confronted with all these different confusions and all the different important parts of the python programming language. Another requirement was that the codes be more difficult than the students were accustomed to. This so that students must apply their problem-solving skills. Through this, this research hopes to discover how students read and understand the code. Further code design choices will be explained with the pieces of code. The code colors and line numbers are only shown in this section because this will make the explanation process easier, the 4 space indents are also replaced by 2 space indents. This means that the students had a black and white version without any line numbering and 4 space indents. Below is the legend for this color coding.

COLOR	MEANING
BLACK	Necessary Code
RED	Unnecessary Code
GREEN	Required prints
BLUE	Necessary variables

Table 1 - Code Legend

6.2.1 CODE 1

```
1   # CODE 1
2
3   temperatuur_in_graden = [20, 5, 4, 15, 14, 0, 17, -2, -5]
4   lekker_warm = 4
5   heel_koud = 14
6
7   nederland = {
8     'cool': 12,
9     'duistland': 41,
10    'holland': 0,
11    'boardgame': -10,
12  }
13
14  for item in temperatuur_in_graden:
15    # item = an element of the list
16    if item <= lekker_warm:
17      print('Lekker zonnen en naar het strand')
18    elif item >= heel_koud:
19      print('Lekker thuis met een kop chocolademelk')
20    else:
21      print('Niet geweldig weer')
22  # wat wordt hier geprint?
```

Figure 2 - C1

Translated code

Figure 3 - C2 Translated

Reasoning and concepts

```
1   # CODE 1
2
3   temperature_in_degrees = [20, 5, 4, 15, 14, 0, 17, -2, -5]
4   warm = 4
5   very_cold = 14
6
7   netherlands = {
8     'cool': 12,
9     'duistland': 41,
10    'holland': 0,
11    'boardgame': -10,
12  }
13
14  for item in temperature_in_degrees:
15    # item = an element of the list
16    if item <= warm:
17      print('Tanning and going to the beach')
18    elif item >= very_cold:
19      print('Delightful at home with a cup of chocolate milk')
20    else:
21      print('Not great weather')
22  # what is printed here?
```

This code has been translated from Dutch to English. Due to this translation, nuances have been lost. The Dutch code had been used during the interviews. This code mainly tries to address RQ 1b, 1d, and RQ 1c. This program tries to imply that when it is cold you can go to the beach and when it is warm you must sit at home with a cup of cholate milk and when it is neither it is not great weather. The variables warm and very_cold contradict common knowledge, 4 degrees Celsius is not nice and warm and 14 degrees Celsius is not cold (lines 4 and 5). Furthermore, between those values, it could still be nice weather, but in this program, this is labeled as 'not great weather' (line 21). This code is thus illogical and tries to answer research question 1b. The code also contains the dictionary of the Netherlands, which is not necessary for the program's printed statements. It also has a spelling mistake in 'duistland', the correct spelling would be 'duitsland or Duitsland' which means Germany. This means that this piece is unnecessary and RQ 1d is addressed. Moreover, it is possible that some students do not know the meaning of the dictionary or the assignment of 'item' within the for-loop and thus RQ 1c could also be answered in this code. These could result in a lack of information where it is unclear or not known if this is a mistake or a genuine feature of the code.

Required knowledge / important parts to successfully complete the code:

- For-loop
- If/else/elif-statement
- Lists
- Dictionaries
- Comments

Generalized execution

First, the variables are created (lines 2-4). Then the variable item is assigned all the items of the 'temperature_in_degrees' list, one by one from start to finish (lines 14-21). If this item is smaller than warm then 'Tanning and going to the beach' is printed (lines 16-17), otherwise if 'item' is bigger than very_cold 'Delightful at home with a cup of chocolate milk' is printed (line 18-19), if none of these are fulfilled then 'Not great weather is printed' (line 20-21).

The expected output would be:

Dutch	English
Lekker thuis met een kop chocolademelk	Delightful at home with a cup of cholate milk
Niet geweldig weer	Not great weather
Lekker zonnen en naar het strand	Tanning and going to the beach
Lekker thuis met een kop chocolademelk	Delightful at home with a cup of cholate milk
Lekker thuis met een kop chocolademelk	Delightful at home with a cup of cholate milk
Lekker zonnen en naar het strand	Tanning and going to the beach
Lekker thuis met een kop chocolademelk	Delightful at home with a cup of cholate milk
Lekker zonnen en naar het strand	Tanning and going to the beach
Lekker zonnen en naar het strand	Tanning and going to the beach

Table 2 - Outcome of C1

Require knowledge / important parts to successfully complete the code:

- It is important to know that item becomes every item in the list
- Know how if/else/elif-statements work
- Know how lists work

6.2.2 CODE 2

Figure 4 - C2

```
1   # CODE 2
2
3   def bbn(a):
4     b = 2
5     c = 0
6     d = 1
7     e = 0
8     while a != c:
9       if (a == 1):
10        e += 1
11        a = a - 1
12      elif (a == 0):
13        e += 0
14      if (b < a and a > b * 2):
15        b = b * 2
16        d += 1
17      elif (b <= a and a < b * 2):
18        a = a - b
19        e += int('1' + '0' * d)
20        b = 2
21        d = 1
22    print(e)
23
24  bbn(6)
```

Reasoning and concepts

This code plays into the lack of processing power and thus the limited space in the working memory. Accordingly, this code mainly tries to answer RQ 1e, but also RQ 1c. RQ 1e is addressed by the large number of different variables (a, b, c, d, e) which are all used, changed, and checked in the code. This means that this code is too large to instantly understand and thus is more complex. RQ 1c is mainly addressed due to the integer multiplication in line 19, this is a special feature of Python. In Python if a string 'b' is multiplied by 2, this means "'b' * 2", this will result in the following string 'bb'. Also '!=' means, is not equal to.

Important concepts used in this piece of code are:

- Functions
- If/else/elif-statement
- While-loop
- String multiplication in Python

- Type conversion (int())
- Boolean statements (!=)

Generalized execution

The function is created, and after this 6 is inputted into the function. This means that 'a = 6' (line 24). In the first loop the third statement is correct and will execute resulting in 'b = b * 2 = 4' and 'd += 1 = 2' (line 8-16). After this 'a' is not equal to zero, thus another loop is started (line 8). During this loop the fourth statement is satisfied and this causes 'a' to equal 'a – b = 2 – 4 = 2' and 'e' to equal 'int('1' + '0' * d) = 100' and b and d become 2 and 1 respectively (line 18-21). Following this, 'a' is still not equal to zero and the fourth statement is again correct meaning, 'a' becomes '2-2 = 0' (line 18), and 1 is added to 'e' (becoming 101). In the following loop, the while statement is not satisfied, consequently, line 22 is executed and '101' is outputted and printed.

Expected output:

110

Required knowledge / important parts to successfully complete the code:

- While loop
- String multiplication
- Many variables

6.2.3 CODE 3

Figure 5 - C3

```
1   # CODE 3
2
3   name = input('What is your name?: ')
4   surname = input('What is your surname?: ')
5
6   mylist = [0,3,1,9,12,5,15,19,2,7]
7
8   true_list = [1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0]
9
10  i = 1
11
12  while i <= 2:
13    mylist.append(int(i*i))
14    i += 1
15
16  mylist.sort()
17
18  mylist[0]
19
20  last = mylist.pop()
21
22  high = max(mylist)
23
24  place = mylist.index(3)
25
26
27  print(mylist)
28  print(last)
29  print(high)
30  print(place)
```

Reasoning and concepts

In this piece of code, the student is tasked with parts of code they have not seen before, this means that they have a lack of knowledge. To make sure that the student does not know one or more parts this code has 5 distinct functions and constructs. This code also adds a couple of lines of unnecessary code. This results in a piece of code that could answer research questions 1c and 1d. The variable names are chosen so that the student could guess their meaning if they do not know what the code does. An important note on this code is that input() technically outputs text to get the

users' input, but this is ignored. Because prints are asked not what is outputted to the command line.

Important concepts used in this piece of code are:

- While-loop
- Lists
- List manipulations (append, pop)
- Index
- Max

Generalized execution

Firstly, 'mylist' is created (line 6). After this 1 and 4 are added by using a while loop and iterating with 'i'. This 'i' first becomes 1, which means that 1*1 is appended to the list. Then 'i' becomes 2, and 4 is added to the list (lines 12-14). In line 16 the list is sorted, this will result in the following list: [0, 1, 1, 2, 3, 4, 5, 7, 9, 12, 15, 19]. Last is defined as 'mylist.pop()' (line 20), so 19 is removed from the list and stored in last. High becomes the highest number in the list, with 19 gone it means that 15 is the highest number (line 22). 'Place' is the index of the number 3 in the list. Number 3 is on the fourth spot in the list (because counting starts at 0), thus place becomes 4 (line 24). Then, 'mylist', 'last', 'high', and 'place' or printed resulting in the following printed output:

Expected output[1]:

```
[0, 1, 1, 2, 3, 4, 5, 7, 9, 12, 15]
19
15
4
```

Required knowledge / important parts to successfully complete the code:

- Knowing the meaning of append, pop, index(), max()
- Lists

[1] With input() ingored

6.3 Interview setup

Location and before the interview

The interview will take place at the school where the student himself/herself is attending. During this interview, only the interviewer and the student will be present. To reassure the students, they are told that they do not have to be an expert in programming. The three pieces of code are printed out and will be shown to the students one by one. Thus, the pieces of code are on paper. This means that the student can also use tools. These tools (pens and markers) are provided by the interviewer. This will also be told to the student. Before the first piece of code is shown to the student, the student will be told that they only need to find what is printed in the program. The student will also be told that the interviewer will not give hints and will only say 'ok' or 'yes'. Finally, the student will be told to read aloud what they are reading. After the explanation, a voice recorder will be started, and the student will be told that this is for analytic purposes only.

During the interview

During the interview, the student will read aloud what they are working on. If this is forgotten it will be asked again, by asking the student to keep reading aloud. This means that the student says what he or she is reading. The interviewer will not say anything while reading the different pieces of code. If the learner would like clarity, only 'yes' or 'ok' will be said, regardless of whether the student was right or wrong.

After the interview

If the student wants it, at the end of each piece of code the interviewer will explain what the correct answer was for that piece of code. The student will also be told not to share the code's content and solution with the other students. When all code snippets have been completed, the student is thanked and the voice recording is stopped. A few more notes will be made by the interviewer about standout elements during the interview.

6.4 Data processing

The data will be processed in a similar method to Wu et al. (2021). The grounded theory (Urquhart, 2013) was also used and the text was also processed into different tables. This thesis will also use a qualitative inductive data analysis. Where inductive data analysis refers to an approach that primarily uses detailed readings of raw data to derive concepts, themes, or a model through interpretations made from the raw data by an evaluator or researcher as described in Thomas (2006).

The information will be gathered through the collected data. The voice recordings are transcribed and labeled. The following labels will be used:

- Belongs to the solving of code 1
- Belongs to the solving of code 2
- Belongs to the solving of code 3
- Belongs to the introduction, explanation, or ending of the interview

These labels are color coded into the interviews such that it is immediately clear what label the corresponding text belongs to. Furthermore, the interviews will have time stamps, noted as [minutes : seconds]. These time stamps will be made when a code is started and will be stopped when a clear answer is given to the corresponding code. Some text might be skipped during the transcription process, this will also be highlighted as { … } where … corresponds to the reason why some spoken words were skipped. If nothing is said for more than 3 seconds during the interviews this will be denoted as … … in the transcript.

The grounded theory (Urquhart, 2013) will be used to further analyze this data. This theory consists mainly of three steps. Open coding, axial coding, and selective coding. This theory is also used in similar research (Wu et al. 2021).

Open coding is done by coding the transcript. This is completed while reading and labeling the transcript of each interview. There have been multiple passes to correctly code the data and make sure that nothing is missing from the recorded data.

Axial coding is then applied by creating categories that the different coded parts fall under. These and some of the open-coded basic labels are processed into tables. These tables include the category, whether and where it occurs in the interview, and further explanation on why this outcome was concluded.

This research will also include tables on how long a student took for the corresponding text. This is done using the timestamps in the text. And another table will be given on how many words the student and the interviewer have said. Any further information will be given underneath any of the tables.

After this, we will apply selective coding to combine various categories into a new theme if possible.

7 RESULTS

In this section, we look deeper into the transcribed interviews. We analyze these interviews and put the gathered data into different tables. Consequently, the results are mainly presented in tables, comments are added to this table and extra comments are added to the explanation below the tables. Each table is filled using the transcribed interviews found in the appendix (section 10.2). In every table C1, C2, and C3 correspond to code piece 1, code piece 2, and code piece 3 respectively. If the given statement is true, then an '✓' will be added to the proper row and column. In the appendix (section 10.2) every interview is visible color coded so that it is clear which code is being discussed. The interviews are separated into different numbers N1 to N13, where N1 is the first interview and N13 is the last interview. Any explanation or remarks are added to the 'Explanation / Remarks' column of the table. These are ordered in such a way that first the code piece is mentioned (C1, C2, or C3) and then a remark or explanation corresponding with that piece of code is given. If no remark or explanation is given that means that the student followed the meaning of the table without any irregularities. Rows will be omitted if no information was in that row.

7.1 RQ 1a - In what order do students read and comprehend code and natural language?

Code read from top to bottom

Interview (N)	C1	C2	C3	Explanation / Remarks
N1	✓	✓	✓	C1, C2 but when a = 6 was found went back into the loop (followed execution flow), C3
N2	✓	✓	✓	C1, C2 read through the code quickly until the student discovered a = 6 then followed the execution flow, C3
N3	✓	✓	✓	C1, C2 read through the code quickly until the student discovered a = 6 then followed the execution flow, C3
N4	✓	✓	✓	C1, C2, C3
N5	✓	✓	✓	C1, C2 after finding a =6 execution flow was followed C3
N6	✓	✓	✓	C1, C2 after reading a = 6 execution flow was followed, C3
N7	✓	✓	✓	C1, C2 code was read only once from top to bottom, C3
N8	✓		✓	C1 read once from top to bottom, C2 code read from top to bottom, but big parts were skipped until print(e) was found, C3
N9	✓		✓	C1, C2 read only what the student could understand this led to a more random reading pattern, C3
N10	✓	✓	✓	C1, C2, C3
N11		✓	✓	C1 quickly scanned the code until the prints, C2, C3
N12	✓	✓	✓	C1, C2, C3
N13	✓	✓	✓	C1, C2, C3
Total (T)	11	10	12	

Table 3 - Code read from top to bottom

Table 3 entails that the code was read from top to bottom. This means that the student read every line from the top to the bottom of the code. This condition needs to be satisfied at the start of reading the code. If any deviation occurs from this reading pattern it will be described in the explanation / remarks box. Note that after reading it once from top to bottom a different reading pattern could become visible. Almost every time the code was read from top to bottom. In C3 this is inevitable as the code is written in such a way that reading it from top to bottom is the same as following the execution flow of the code. C2 and C1 have some deviations from this top to bottom reading pattern. These occurred in C1 during N8 and N9, where the code was read more randomly, or big parts were skipped. C1 was only once different, namely in interview N11. During this interview, the student quickly scanned the code and skipped big parts of it to find the prints. It is interesting to note that this pattern was not found during C2 or C3.

C2 is also interesting, most students read it from top to bottom, but after they finished reading the code and gathered the required information a significant proportion of the students read the code following the execution flow (N1, N2, N3, N5, N6, N7). From **Table 3** we can see that S2 from these interviews in C2 where on average higher than the average score of N4, N8 N9, N10, N11, N12, and N13. Namely, a 0.68

The total score table and breakdown of the table will be discussed in section 7.6.1. Only one student in this research used scanning of the code until the prints. This also only happened at code piece 1. According to the teacher, this student did well in the programming exercises but did not excel in them. The score of this student's S1 was a 1, so this student understood the code correctly.

In conclusion, almost all students read the code from top to bottom. Some deviated from this reading pattern after reading it once, these students also performed better than those that did not apply this reading strategy. Only one student quickly scanned the code and looked for a solution, but their performance was not notably better.

7.2 RQ 1b - How do students read illogical code (where illogical code contradicts common knowledge)?

Made a note on illogical code

Interview (N)	C1	Explanation / Remarks
N1	✓	NL: "interessant, normaal zou het omgekeerd zijn" EN: interesting, normally it would be the other way around
N2	✓	NL: "Deze snap ik niet waarom wordt er als er kleiner of gelijk is" EN: This one I don't understand why, if there is less than or equal to
N3	✓	NL: "Hoe lager het is, is dan blijkbaar goed" EN: The lower it is, then apparently is good
N6	✓	NL: "Gek dat die andersom zijn dan he" EN: Crazy that they are the other way around then
N7	✓	NL: "dit zijn dan geen temperaturen volgens mij" EN: these are not temperatures in my opinion
N9		C2 made a note that the names did not have anything to do with the rest of the code
N12	✓	NL: "Als het warm, wacht huh … Het is een rare code want het kan aan mij liggen want koud is kleiner dan warm" EN: When it's hot, wait huh ... It's a weird code because it could be just me because cold is smaller than hot
Total (T)	6	

Table 4 - Notes on illogical code

A little less than half of the interviews mentioned the illogical code context of C1 (N1, N2, N3, N6, N7, N12). Most students find it crazy or interesting. Some students begin to question the names of the variables (N7). Only one student used the incorrect more logical code, but after spending some more time reading the code the student changed their answer. The students that mentioned the illogical code did not get a significantly better answer or a worse answer with an average of S1 of 0.65 and the total average of all S1 being 0.55. Only one student (N12) made a note on the incorrectly spelled 'duistland'.

N1 and N10 used a different variable than temperature_in_degrees. They used the netherlands dictionary. Where they used the values of the dictionary instead of the values of the list.

N1 responded to this saying that it looked very important and N10 was not asked why they used this dictionary. It could be because of the confusion of using temperatures that are not according to common knowledge thus using a different list that is not temperatures would be more logical. Or it could be that they did not understand how to use the list and thus tried it with the dictionary. Although, currently unclear why the students choose this option, it would be interesting to investigate further.

In conclusion, most students notice and mention the illogical code, but they did not change their answers because of this. Sometimes, incorrect extra code was used to get an answer.

7.3 RQ 1c - How do students read code whose functions and methods they have never seen before?

Created own meaning for unknown variables

Interview (N)	C1	C2	C3	Explanation / Remarks
N1	✓		✓	C1 item was an item of Netherlands dictionary C3, sort was guessed as sorting from small to large, filled in for index(3), wrongly namely mylist[3],
N2			✓	C3 guessed the meaning of sort correctly
N3			✓	C3, place = index(3) seen as mylist[3], no guess made for sort, was not clear how it would be sorted
N4			✓	mylist[0] was given new meaning as mylist => [0], high was guessed but after the guess was retracted, sort was guessed, place is the first 3 numbers of the list
N5			✓	C3 place was given mylist[3], pop was guessed by last
N6			✓	C3 saw pop us population, thus the number of items in the list, place was given mylist[3] sort was guessed from small to big
N7		✓	✓	C2, Created own meaning to a, C3 place as mylist[3] and guessed pop based on last
N8			✓	C3 append as removing something from the list, not stated what, guessed sort as being sorted from big to small, place is mylist[3], pop is being seen as last due to the name
N9			✓	C3 max was guessed based on the name
N10	✓		✓	C1 item is of the netherlands dictionary C3 max guessed based on name, place was guessed as mylist[3]
N11	✓		✓	C1 item was a random item in the list, C3 place is mylist[3]
N12		✓	✓	C2 filled a = 0 in, C3 filled in last number for last in sorted list, place index(3) was thought as mylist[2], the highest number was guessed by the name high and max
N13	✓		✓	C1 item was a random item in the list, C3 place is mylist[3] and high is the maximum of the list
Total (T)	3	2	13	

Table 5 - Created own meaning

C3 was mainly written to answer this research question, but due to the difficulty level of C1 and C2, they also give insight into this research question. It is visible that everyone guessed some of the meaning for variables in C3. Max was always guessed (13 times), sort and index/place were almost always guessed (sort skipped in N3, N5, N11, guessed 10 times out of 13 and index/place skipped in N2, N9, guessed 11 times out of 13) but pop/last was seldom guessed (pop/last guessed in: N3, N5, N6, N7, N8, guessed 5 times out of 13). Append was only guessed in N8.

In C2 it was mostly unclear what the ‘a’ stood for as it was not clearly declared (for example ‘a = 8). Two students (N7, N12) created their own meaning for ‘a’ in C2. N12 filled 0 for a and the other student was not clear on what they did exactly to ‘a’.

In C1 it was mainly unclear what the item was supposed to be so. Two students (N11, N13) guessed that item would become a random value of the chosen list. Another two students (N1, N10) guessed that item would become a value of the netherlands dictionary.

Skipped necessary code

Interview (N)	C1	C2	C3	Explanation / Remarks
N1		✓	✓	C2 skipped while loop, C3 skipped last = mylist.pop and skipped append, did not know the meaning
N2		✓	✓	C2 skipped while loop, C3 skipped append, pop and place = index
N3				
N4		✓	✓	C2, skipped bbn(6), thus skipped the whole while loop, C3 skipped pop,
N5		✓	✓	C2 skipped while loop, C3 skipped sort
N6				
N7	✓	✓	✓	C1 skipped for loop, C2 skipped bbn(6), C3 skipped append
N8		✓	✓	C2 skipped while loop, C3 skipped print(mylist)
N9	✓	✓		C1 skipped almost all code so only made a conclusion based on what the student knew, C2 skipped almost all code
N10		✓	✓	C2 skipped almost all code, C3 skipped almost all code
N11		✓	✓	C2 skipped bbn(6) did not know what to do, C3 skipped print(mylist), skipped append
N12		✓		C2 did not fill in a so used ‘a = 0’
N13		✓	✓	C2 skipped the entire loop because ‘a’ was not given, C3 skipped append
Total (T)	2	10	10	

Table 6 - Skipped code

Some students skipped the code that they did not understand. In the above table, some parts were skipped but clearly, they had seen them before so these will be omitted in this section of the analysis. Student N9 skipped almost all code in C1 and in C2 and student N10 skipped almost all code in C2 and in C3. Student N10 mentioned that it was too difficult to understand and overwhelming. A lot of students skipped bbn(6) in C2, because this was not understood how it exactly works, namely at least students N4, N11, N12, and N13. Some students tried to create their own ‘a’ by going through the function.

In C3, all the parts that were not guessed in the previous table were skipped by the students. And in C1 N7 skipped the for loop.

In conclusion, most students try to guess what the variables mean even though they do not have the required knowledge to correctly execute the code. The guesses change depending on the person, if a student was more confident, they were more likely to guess what the variables mean. Most guessed variables are derived from their name. If the student understood the meaning of both the function and the variable name, the name was always guessed. But if only one of both was understood, for example: 'pop' or 'place' then they would sometimes guess the meaning. If nothing was understood or partially understood, for example: 'append', then nothing was guessed. If a student did not understand a part of the code they would skip it. If it was clear that a part was important to the solution, like 'a' in C2, sometimes students would create their own 'a' by using the context of the if-statements.

7.4 RQ 1d - How do students deal with unnecessary pieces of code?

Used unnecessary code

Interview (N)	C1	C3	Explanation / Remarks
N1	✓		C1 used the Netherlands dictionary instead of the temperatures_in_degrees list
N4		✓	C3, mylist[0] thought that it would transform this list to [0]
N10	✓		C1 used the Netherlands dictionary instead of the temperatures_in_degrees list
N13		✓	C3 thought that mylist[0] would change the entire list to [0]
Total (T)	2	2	

Table 7 - Used unnecessary code

In the above Table 7, it is clear that not many people did something special with the unnecessary code. Student N9 made a note on code C2 that the names did not have anything to do with the rest of the code. Two students (N1, N10) used the Netherlands dictionary instead of the temperatures_in_degrees list. As described in section 8.2 N1 responded to this by saying that it looked very important and N10 was not asked why they used this dictionary. It could be because of the confusion of using temperatures that are not according to common knowledge thus using a different list that is not temperatures would be more logical. Or it could be that they did not understand how to use the list and thus tried it with the dictionary. Although, currently unclear why the students choose this option, it would be interesting to investigate further.

In code C3, mylist[0] was unnecessary for the completion of the code. Two students (N4, N13) used this code incorrectly, however. They both thought that mylist[0] would change the entire list to [0]. Changing mylist to [0] means that none of the following code would be correct.

In conclusion, some unnecessary code was used. This was mainly done when the correct meaning or usage of that piece of code was not understood. If this was the case, sometimes the meaning of the code was guessed and this was done mostly incorrectly.

7.5 RQ 1e - How do students deal with complex code (code with a lot of variables)?

Used tools; made notes on the code

Interview (N)	C1	C2	C3
N5		✓	
N6		✓	✓
N7		✓	✓
N9	✓	✓	✓
N10	✓	✓	✓
N13		✓	✓
Total (T)	2	6	5

Table 8 - used tools

In the above Table 8, we can see that some students used tools / made notes on the code in order to understand this correctly. Most students made notes on code C2, but C3 is not far behind C2. Two students made notes on C1. All notes were made with numbers, to remember what the value of the corresponding variable was.

Skipped necessary code

Interview (N)	C2	Explanation / Remarks
N1	✓	C2 skipped while loop
N2	✓	C2 skipped while loop
N4	✓	C2 skipped bbn(6)
N5	✓	C2 skipped while loop
N7	✓	C2 skipped bbn(6)
N8	✓	C2 skipped while loop
N9	✓	C2 skipped almost all code
N10	✓	C2 skipped almost all code
N11	✓	C2 skipped bbn(6)
N13	✓	C2 skipped the entire loop because ‘a’ was not given
Total (T)	10	

Table 9 - Skipped code

Almost all students that found or filled in an ‘a’ skipped the wile loop in C2. Most of the other students skipped bbn(6) and thus did not find ‘a = 6’ and sometimes did not give an answer at all or they gave an incorrect answer (mainly ‘e = 0’, so 0 is printed). N9 and N10 skipped almost the entire C2 code, saying it was overwhelming and too difficult.

In conclusion, about half the students wrote something down on any of the codes. This was mainly used for remembering what the variable contained. It is noticeable that almost all students that found or filled

in ‘a’ skipped the while loop in C2. Due to the sheer number of variables, some students quickly gave up on solving the code.

7.6 Extra overview

This section contains some extra information that does not directly help in answering the research question but can give some insight and comparison measures between students.

7.6.1 GENERAL PERFORMANCE OVERVIEW

To get a general performance indicator this research has created several indicating factors of the comprehension level of the code. C1 or C2 or C3 correspond to the piece of code that is evaluated under different conditions. If these conditions are met a ‘✓’ is placed inside of the table in the corresponding row and column. S1 or S2 or S3 is the total score of the student, the higher the score the better the understanding of the student. The scores are calculated by the number of fulfilled conditions divided by the total amount of conditions. The explanation of why the given score is scored can be found in the explanation box of the table. If nothing is notable or if the student did not differ at all from the condition no explanation is given.

C1 Legend

A = if statements correct

B = used correct list

C = used item loop correctly

S1 = Total Score of code 1

Read correctly

	C1				
Interview (N)	**A**	**B**	**C**	**S1**	**Explanation**
N1	✓			0.3	The student understood while loop correctly, but instead of using data from temparatures_in_degrees they used the holland dictionary
N2	✓			0.3	The student did not understand 'for item in list', thought only one item would get used and that this item is unknown
N3	✓	✓	✓	1	
N4	✓	✓		0.7	The student thought that there would be an item randomly chosen from the list temperature_in_degrees
N5	✓			0.3	The student understood if statements but did not understand where item came from and did not attempt a guess
N6	✓	✓	✓	1	
N7	✓			0.3	The student did not think that item would become anything, thus the answer was that the code printed nothing
N8	✓			0.3	The student did not fill in item with anything
N9				0	Followed the code by filling it in in a logical manner, such that 4 degrees is at home with a cup of hot chocolate, this should have been at the beach
N10	✓		✓	0.7	Filled using the dictionary
N11	✓	✓	✓	1	The student used item as random item from the list, otherwise, it was correct
N12	✓	✓	✓	1	Correctly understood
N13	✓			0.3	Code thought to use a random item
Total (T)	12	5	5		

Table 10 - C1 Legend and performance metrics

C2 Legend

D = used a = 6

E = correctly used while loop

F = If statement followed correctly

S2 = Total Score of code 2

Read Correctly

	C2				Explanation
Interview (N)	**D**	**E**	**F**	**S2**	
N1	✓		✓	0.7	
N2	✓		✓	0.7	The student did not follow the while loop
N3	✓	✓	✓	1	Student did not correctly do 'int('1' + '0' * d)'
N4			✓	0.3	6 was not inputted into the function, thus thorough understanding bit unclear
N5	✓		✓	0.7	While loop was forgotten,
N6	✓	✓		0.7	While loop was not forgotten but if statements were incorrectly handled so the student thought It was in an infinite loop
N7			✓	0.3	The student did not know what 'a' was, so created their own 'a'
N8	✓		✓	0.7	If statements are followed correctly but the while loop is forgotten
N9				0	Did not know how to follow the code at all
N10				0	The code was too difficult to understand and gave up quickly
N11			✓	0.3	If statements were followed correctly but 'a = 6' was not inputted. Also, the student told the while loop correctly but later incorrectly corrected the while loop usage, the student's answer was correct, but they backtracked
N12			✓	0.3	'a = 6' was not found so 'a = 0' was used
N13				0	Skipped code because a was never given
Total (T)	6	2	9		

Table 11 - C2 legend and performance metrics

C3 Legend

G = Append correctly
H = explained while loop correctly
I = Sort correctly
J = Last correct
K = high correct
L = place correct
M = mylist correct
S3 = Total Score of code 3

Read correctly

	C3								
Interview (N)	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**S3**	**Explanation**
N1		✓	✓		✓		✓	0.57	High was guessed, mylist pop skipped, index(3) seen as mylist[3], skipped append
N2			✓		✓		✓	0.43	
N3				✓	✓			0.29	
N4		✓	✓					0.29	Thought that mylist would become [0] after mylist[0] thus further code was incorrect
N5		✓		✓	✓		✓	0.57	
N6		✓			✓		✓	0.43	
N7		✓	✓		✓		✓	0.57	Pop correctly guessed, but after the guess was changed incorrectly
N8				✓	✓			0.29	Mylist print is forgotten to be mentioned
N9					✓			0.14	Only the variables that sounded logical were understood the rest was skipped
N10		✓						0.14	Code was explained what the person knew, but not much more
N11		✓	✓		✓			0.43	
N12		✓	✓	✓	✓		✓	0.71	The code was done almost correctly
N13		✓	✓		✓		✓	0.57	Thought that mylist[0] would change the entire list to [0]
Total (T)	0	9	7	4	11	0	7		

Table 12 - C3 legend and performance metrics

7.6.2 TIME TAKEN AND WORDS SAID

This section contains some interesting extra data. This is broken down into 4 different tables. Table 13 and Table 14 contain the number of words said during the interview and Table 14 also contains the score (S1, S2, S3) of the student. Table 15 and Table 16 hold the time it took to complete the code. Table 16 also contains the score (S1, S2, S3) of the student.

	Total Words said			Words said by interviewer			Words said by interviewee		
Number(N)	**code 1**	**code 2**	**code 3**	**code 1**	**code 2**	**code 3**	**code 1**	**code 2**	**code 3**
N1	193.00	324.00	297.00	55.00	65.00	28.00	138.00	259.00	269.00
N2	335.00	140.00	215.00	52.00	27.00	34.00	283.00	113.00	181.00
N3	178.00	372.00	206.00	36.00	12.00	22.00	142.00	360.00	184.00
N4	280.00	248.00	272.00	69.00	54.00	19.00	211.00	194.00	253.00
N5	216.00	234.00	145.00	62.00	14.00	21.00	154.00	220.00	124.00
N6	243.00	219.00	274.00	10.00	17.00	10.00	233.00	202.00	264.00
N7	160.00	211.00	187.00	22.00	15.00	35.00	138.00	196.00	152.00
N8	165.00	312.00	163.00	22.00	20.00	3.00	143.00	292.00	160.00
N9	203.00	129.00	268.00	77.00	35.00	80.00	126.00	94.00	188.00
N10	252.00	284.00	226.00	39.00	33.00	21.00	213.00	251.00	205.00
N11	239.00	317.00	216.00	50.00	14.00	41.00	189.00	303.00	175.00
N12	286.00	299.00	193.00	24.00	11.00	4.00	262.00	288.00	189.00
N13	214.00	93.00	294.00	27.00	12.00	16.00	187.00	81.00	278.00
Average:	228.00	244.77	227.38	41.92	25.31	25.69	186.08	219.46	201.69
St. Dev.	…	…	…	…	…	…	51.46	85.15	49.10

Table 13 - Words said

The above Table 13 displays how many words have been said in total, how many words have been said by the interviewer, and how many words have been said by the interviewee (the student). This has been done by counting the words said in the transcribed data. The icons show the highest (▲), the lowest (▼), and the middle value (▬) of the corresponding word count. This would mean that student N1 said the most words in C3 (▲) and the least words in C1 (▼). The table also has a bar in C1, C2, and C3 in blue, green, and red colors. The fuller the bar is the higher the value compared to the other values in this column. So, this bar compares the different students on the number of words said per code.

We can see that on average C2 was the code where the most words were said and that C1 is the code where on average the least number of words were said. This somewhat corresponds to the number of executed lines of code in each code. Although following that rationale it would mean that C3 had the average smallest number of words said, but this is not true.

Words said by student vs Performance scores						
Number(N)	C1	S1	C2	S2	C3	S3
N1	138.00	0.33	259.00	0.67	269.00	0.57
N2	283.00	0.33	113.00	0.67	181.00	0.43
N3	142.00	1.00	360.00	1.00	184.00	0.29
N4	211.00	0.67	194.00	0.33	253.00	0.29
N5	154.00	0.33	220.00	0.67	124.00	0.57
N6	233.00	1.00	202.00	0.67	264.00	0.43
N7	138.00	0.33	196.00	0.33	152.00	0.57
N8	143.00	0.33	292.00	0.67	160.00	0.29
N9	126.00	0.00	94.00	0.00	188.00	0.14
N10	213.00	0.67	251.00	0.00	205.00	0.14
N11	189.00	1.00	303.00	0.33	175.00	0.43
N12	262.00	1.00	288.00	0.33	189.00	0.71
N13	187.00	0.33	81.00	0.00	278.00	0.57
Average:	186.08	0.56	219.46	0.44	201.69	0.42
St. Dev.	51.46	0.34	85.15	0.32	49.10	0.18

Table 14 - Words said and scores

In the above Table 14, the performance scores (S1, S2, S3) have been added. And they are also accompanied by bars within the cells, these bars compare the values of the column with each other. The more filled the bar the higher the number compared to other numbers. In C1 the more words the student has said the higher the score is that they received. But this cannot be said for C2 and C3, there the difference is not significant enough.

Time for each piece of code									
Number(N	Start-1	End-1	Start-2	End-2	Start-3	End-3	C1	C2	C3
N1	00:05	02:16	03:05	06:06	07:00	09:48	02:11	03:01	02:48
N2	00:40	03:18	03:38	05:10	05:25	08:09	02:38	01:32	02:44
N3	00:48	02:40	02:50	09:12	06:20	08:20	01:52	06:22	02:00
N4	00:40	03:05	03:15	05:00	05:05	07:00	02:25	01:45	01:55
N5	00:40	03:40	03:45	06:22	06:30	08:35	03:00	02:37	02:05
N6	00:43	03:15	04:00	06:50	07:40	10:55	02:32	02:50	03:15
N7	00:50	02:30	03:30	06:25	07:05	09:30	01:40	02:55	02:25
N8	00:41	03:15	04:07	08:11	09:05	11:22	02:34	04:04	02:17
N9	00:39	03:15	04:10	05:45	06:40	10:05	02:36	01:35	03:25
N10	00:50	04:20	05:42	09:30	10:15	13:22	03:30	03:48	03:07
N11	00:49	02:50	03:10	07:20	08:10	10:25	02:01	04:10	02:15
N12	00:30	03:20	04:05	06:40	08:10	10:20	02:50	02:35	02:10
N13	00:30	03:08	03:45	05:00	05:40	09:04	02:38	01:15	03:24
Average:	00:38	03:08	03:46	06:43	07:09	09:45	02:29	02:57	02:36

Table 15 - Time taken

The above Table 15 displays the time taken for the different parts of code during the interview. The bars and arrows have the same meaning as in Table 13. This table shows that on average people take the most amount of time at code 2 and the least amount of time on code 1. Although the differences between these times are minimal (standard deviation of 1:37). It is interesting to note that almost all interviews lasted around 9:45 minutes (see column END-3) even though no time limit was given. This time does include the explanation, without the explanation time the interviews are also quite close in variation. None of the values are significant enough to say that there is a consistent average time.

Time vs score						
Number(N)	**C1**	**S1**	**C2**	**S2**	**C3**	**S3**
N1	02:11	0.33	03:01	0.67	02:48	0.57
N2	02:38	0.33	01:32	0.67	02:44	0.43
N3	01:52	1.00	06:22	1.00	02:00	0.29
N4	02:25	0.67	01:45	0.33	01:55	0.29
N5	03:00	0.33	02:37	0.67	02:05	0.57
N6	02:32	1.00	02:50	0.67	03:15	0.43
N7	01:40	0.33	02:55	0.33	02:25	0.57
N8	02:34	0.33	04:04	0.67	02:17	0.29
N9	02:36	0.00	01:35	0.00	03:25	0.14
N10	03:30	0.67	03:48	0.00	03:07	0.14
N11	02:01	1.00	04:10	0.33	02:15	0.43
N12	02:50	1.00	02:35	0.33	02:10	0.71
N13	02:38	0.33	01:15	0.00	03:24	0.57
Average:	02:29	0.56	02:57	0.44	02:36	0.42

Table 16 - Time taken and scores

The above Table 16 adds the different performance scores to the time taken table. The bars have the same meaning as in the previous Table 15. On average there is not something interesting in C1, S1 and C3, S3, but C2 shows that the lower the time taken the less likely it is to get a higher score.

In conclusion, the time taken generally follows the number of lines that need to be executed to finish the code. If a student deviates from this it is likely that they scored lower. Words said follow a similar pattern. Most interviews lasted around 9:45 minutes.

8 DISCUSSION

The results show that almost all of the students read the code from top to bottom. This reading pattern deviated sometimes after reading through the code once. The better the understanding of the code the more strictly the execution flow was followed. One time the code was quickly scanned for the printed pieces of code in an effort to follow the program execution. According to Busjahn et al. (2015), this reading order is more likely to be seen in more experienced programmers. This was later discussed with the teacher, they said that this student performed well during coding tasks, but not exceptionally well. The student was still a novice. So this is an exception from the expected reading order of novice programmers. The rest of the read patterns (top to bottom reading / linearly) follow the code reading order as described in Busjahn et al. (2015) that more novice programmers express. This was expected. When the students would not understand most code, the student would mainly scan the code from top to bottom and explain to the interviewer what they did understand. This could be seen in the transcript as the student did not talk for a long time at the start of the interview. This behavior was also seen in a research done by Wu et al. (2019)

The illogical code did not hinder most students. Two students used different code (Netherlands dictionary instead of the temperatures_in_degrees list), this could be because of this illogical code. One of the students said that the code looked important as an explanation as to why the student used the different pieces of code. The other student was not asked why they choose to use the wrong data. An explanation for this could be that they were overwhelmed by the code and thus choose the easier to follow dictionary or that they did not understand the illogical part and thought that the random numbers in the netherlands dictionary were more rational to use than the temperatures. Or it could be a combination of both cases. Almost all students noticed the illogical code and made a note of it. This could be to try to confirm the finding with the interviewer.

Most unknown variables were guessed if the variable names were somewhat clear. This makes sense if a student does not know at all what the words would mean. There is no point in guessing. The function name was more often used in guessing the variable. This would mean that if the student did not understand the name of the function they are more likely to stop guessing the meaning of the variable than not knowing the name of the variable. This corresponds with the fact that Index() was guessed more than Pop(). This even though ‘last’ is probably more informative than ‘place’. The students also gave up on guessing earlier if they did not understand any of the surrounding code. This is not only true for the guessing part but also for trying to solve the code in general.

It was also interesting that mylist[0] was identified twice as meaning that it changed the list to mylist = [0]. Even though the later code could not be understood with this change some students stuck to this reasoning or gave multiple options (with mylist[0] and without). This could have been done because of

the previous illogical code. Students could have thought that the previous code (C1) was illogical and this second code could also be illogical.

Most students skipped the while loop in code 2 (C2). This could correspond with SOURCE, where there is not enough space in memory to store all the variables and to not forget the loop. In both code 1 (C1) and code 3 (c3) the while and for loop were mostly not forgotten. Writing it down did not seem to help, this could be due to some form of time pressure or due to students not wanting to rely on their written variables. We can conclude from this that it takes a whole lot of effort to go through code without forgetting a part, especially when so many variables are declared.

Time taken varied, but generally followed the number of lines of code. The more text there is to read or execute the longer the explanation is expected to take. Interestingly, most interviews lasted around 9:45 even though no clock was visible or available for the students. This could be because students saw how long the other students took at the tests and the interviewer could also have had some bias towards taking 10 minutes because this was communicated with the teacher.

8.1 Conclusion

Programming is becoming increasingly important. Reading code is an important skill to learn programming. Reading code improves other programming skills. But not much research has been done on how high school students read code. This research explored how high school students read and comprehend code. This studys confirmed that most novice programmers read code in a similar way as reading natural language (top to bottom). This was mainly true whilst reading through the code for the first time. After this first time, the reading order sometimes deviated from this and followed the execution flow. The illogical code was not followed by the students, although it may have resulted in some indirect mistakes. Almost all students made a note on the illogical code. Most functions and methods that the students did not understand were guessed. It depended mainly on the name of the function or method, but also the name of the variable, whether an attempt was made to guess the meaning. Sometimes unnecessary pieces of code were used to complete the code. This despite sometimes introducing errors in the rest of the code. Explanations were given as to why these unnecessary pieces were used, but a concrete conclusion cannot be taken. Code was skipped if the code had too many variables or parts that needed to be stored in memory. This even though the skipped code was essential to successfully completing the code. It could mean that exactly following or understanding code with many functions, loops, and variables is a very difficult or taxing problem.

8.2 Limitations

In this thesis, only a small number of high school students were interviewed. Furthermore, all of the high school students came from the same high school. Additionally, there is a large amount of choice in the content of the Dutch computer science course for high schools. These courses are relatively new and therefore there is not a standard learning procedure yet. This means that the computer science courses from one school could focus more on subject A whilst another school mainly focuses on subject B. There are some standards when concerning the final exam and teaching material. There are a couple of subjects that must be taught. Apart from these obligatory subjects, there are choice subjects, these subjects can be chosen by the teacher and thus can create a difference in what is taught. This all means that the data collected might not be representative of all high school students.

Furthermore, it was sometimes difficult to know what the students were struggling with. If during the interview the student did not disclose all information then we won't have that outcome in this research. Students may have found it difficult to verbalize their thoughts. This is inherently more difficult with the think-aloud interview style. Another implication of the think-aloud interview style is that the students might not have their full attention to solving and reading the code. This is because reading the text out loud also requires some processing power and memory of the student. This could result in worse problem-solving performance or forgetfulness whilst solving the code.

Finally, the interview process could be made more rigid. Rarely students were given options that other students did not have (forgetting in the introduction to say that they could use the provided tools to write on the code). Sometimes some explanation was not correctly understood by some students, so during the interview process, this was improved. But this means that there are (minimal) differences between the interview process of student N1 and student N13. This could result in some unplanned inter-subject deviation.

8.3 Future research

This research was a broad look at the reading process of high school students. Some more detailed studies could be done to get a better understanding of the reading and code comprehension process. For example: how students deal with complex code would be an interesting topic to study further. More research could be done on the indirect mistakes due to illogical code. This research could encompass the effects of illogical code on the rest of the code and next code pieces. Thus finding the importance of logical code for students, whilst it does not matter for the computer executing the code.

Further research could also be done with more high schools and more students. Although this could improve reliability I expect that due to the large difference in high school computer science classes it would be difficult to get a statistically significant outcome. Another interesting topic is if the skills or part of the skills for code reading and normal reading could be transferrable. This would mean that a more skilled reader could be better at reading code.

Although this research discussed working memory, long-term, short-term memory, and a more technical method of reading (phonemes and graphemes). The actual involvement of these concepts within code reading could be researched further.

8.3.1 A NOTE TO HEDY

Hedy (Leiden University, n.d.) is a programming code developed by the Perl research group and is led by my supervisor. Hedy is developed to learn programming and it does this by slowly introducing new concepts. To finish this research I wanted to give some insights on what would be interesting to implement into Hedy when it comes to code reading. I believe that it is important to introduce more complex code early. Where complex code is code with many different variables or at least more than the student is normally accustomed to. This is so that the students can get used to this difficult-looking code and create the discipline required to successfully tackle the problem.

9 BIBLIOGRAPHY

Antropova, M., Shchekotova, G., Begel, A., Hansen, M., Goldstone, R. L., Lumsdaine, A., Ihantola, P., Menzel, S., Orlov, P. A., Sharif, B., Bandarupalli, S., & S. (2013). Eye Movements in Programming Education: Analyzing the Expert's Gaze. School of Computing, UEF, Joensuu, Finland. http://www.emipws.org/assets/emip2013_report.pdf

Atkinson, R., & Shiffrin, R. (1968). Human Memory: A Proposed System and its Control Processes. Psychology of Learning and Motivation, 89–195. https://doi.org/10.1016/s0079-7421(08)60422-3

Ayres, P., & Paas, F. (2009). Interdisciplinary Perspectives Inspiring a New Generation of Cognitive Load Research. Educational Psychology Review, 21(1), 1–9. https://doi.org/10.1007/s10648-008-9090-7

Baddeley, A. (1982). Reading and working memory. Visible Language, 18(4), 414–417. https://www.proquest.com/docview/1297965490?pq-origsite=gscholar&fromopenview=true&imgSeq=1

Bednarik, R., & Tukiainen, M. (2006). An eye-tracking methodology for characterizing program comprehension processes. Proceedings of the 2006 Symposium on Eye Tracking Research & Applications - ETRA '06. https://doi.org/10.1145/1117309.1117356

Begel, A. (2015). Applying Cognitive Theories to Novice Programmers. Free University Berlin. http://www.emipws.org/assets/emipe2_report.pdf

Blascheck, T., & Sharif, B. (2019). Visually analyzing eye movements on natural language texts and source code snippets. Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications. https://doi.org/10.1145/3314111.3319917

Burns, M. K., Davidson, K., Zaslofsky, A. F., Parker, D. C., & Maki, K. E. (2017). The Relationship Between Acquisition Rate for Words and Working Memory, Short-Term Memory, and Reading Skills: Aptitude-by-Treatment or Skill-by-Treatment Interaction? Assessment for Effective Intervention, 43(3), 182–192. https://doi.org/10.1177/1534508417730822

Busjahn, T., Bednarik, R., Begel, A., Crosby, M., Paterson, J. H., Schulte, C., Sharif, B., & Tamm, S. (2015). Eye Movements in Code Reading: Relaxing the Linear Order. 2015 IEEE 23rd International Conference on Program Comprehension. https://doi.org/10.1109/icpc.2015.36

Busjahn, T., & Schulte, C. (2013). The use of code reading in teaching programming. Proceedings of the 13th Koli Calling International Conference on Computing Education Research - Koli Calling '13. https://doi.org/10.1145/2526968.2526969

Busjahn, T., Schulte, C., & Busjahn, A. (2011). Analysis of code reading to gain more insight in program comprehension. Proceedings of the 11th Koli Calling International Conference on Computing Education Research - Koli Calling '11. https://doi.org/10.1145/2094131.2094133

Daneman, M. (1991). Working memory as a predictor of verbal fluency. Journal of Psycholinguistic Research, 20(6), 445–464. https://doi.org/10.1007/bf01067637

Diamond, A. (2013). Executive Functions. Annual Review of Psychology, 64(1), 135–168. https://doi.org/10.1146/annurev-psych-113011-143750

Ehri, L. C. (2005). Learning to Read Words: Theory, Findings, and Issues. Scientific Studies of Reading, 9(2), 167–188. https://doi.org/10.1207/s1532799xssr0902_4

Free University Berlin, Busjahn, T., Schulte, C., Tamm, S., & Bednarik, R. (2015, March). Eye Movements in Programming Education II: Analyzing the Novice's Gaze. Free University Berlin. http://www.emipws.org/assets/emipe2_report.pdf

Guéhéneuc, Y. G. (2006). TAUPE. Proceedings of the 2006 Conference of the Center for Advanced Studies on Collaborative Research - CASCON '06. https://doi.org/10.1145/1188966.1188968

Hermans, F. (2021). The Programmer's Brain. Manning.

Konandur, S., Aniche, M. F., Vargas, E. L., Zaidman, A. E., & Scharenborg, O. E. (2020). Test Code Comprehension: Insights from an Eye Tracker. TU Delft. http://resolver.tudelft.nl/uuid:6ba7bd64-478d-4c44-8a96-78922122cbd9

Leiden University. (n.d.). Hedy - A gradual programming language. Hedy. Retrieved June 23, 2022, from https://www.hedycode.com/

Lister, R., Fidge, C., & Teague, D. (2009). Further evidence of a relationship between explaining, tracing and writing skills in introductory programming. Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education - ITiCSE '09. https://doi.org/10.1145/1562877.1562930

Lister, R., Seppälä, O., Simon, B., Thomas, L., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J. E., & Sanders, K. (2004). A multi-national study of reading and tracing skills in novice programmers. Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education - ITiCSE-WGR '04. https://doi.org/10.1145/1044550.1041673

Muter, V., & Snowling, M. (1998). Concurrent and Longitudinal Predictors of Reading: The Role of Metalinguistic and Short-Term Memory Skills. Reading Research Quarterly, 33(3), 320–337. https://doi.org/10.1598/rrq.33.3.4

O'Shaughnessy, T. E., & Lee Swanson, H. (2000). A Comparison of Two Reading Interventions for Children with Reading Disabilities. Journal of Learning Disabilities, 33(3), 257–277. https://doi.org/10.1177/002221940003300304

Overheid. (2014, January). Inrichtingsbesluit WVO. Wetten Overheid. https://wetten.overheid.nl/BWBR0005946/2014-01-01

Peng, P., Barnes, M., Wang, C., Wang, W., Li, S., Swanson, H. L., Dardick, W., & Tao, S. (2018). A meta-analysis on the relation between reading and working memory. Psychological Bulletin, 144(1), 48–76. https://doi.org/10.1037/bul0000124

Pollock, E., Chandler, P., & Sweller, J. (2002). Assimilating complex information. Learning and Instruction, 12(1), 61–86. https://doi.org/10.1016/s0959-4752(01)00016-0

SLO. (2014, April). Informatica in de bovenbouw havo/vwo. SLO - Nationaal Expertisecentrum Leerplanontwikkeling. https://slo.nl/publish/pages/2840/informatica-in-de-bovenbouw-havo-vwo.pdf

SLO. (2016). Examenprogramma informatica havo/vwo. https://www.examenblad.nl/examenstof/informatica-havo-en-vwo-3/2022/f=/examenprogramma_Informatica_havo-vwo.pdf

Someren, M. W., Barnard, Y. F., & Sandberg, J. A. C. (1994). The think aloud method: A practical approach to modelling cognitive processes. Londen: Academic Press. https://hdl.handle.net/11245/1.103289

Sweller, J., van Merrienboer, J. J. G., & Paas, F. G. W. C. (1998). Cognitive Architecture and Instructional Design. Educational Psychology Review, 10(3), 251–296. https://doi.org/10.1023/a:1022193728205

Thomas, D. R. (2006). A General Inductive Approach for Analyzing Qualitative Evaluation Data. American Journal of Evaluation, 27(2), 237–246. https://doi.org/10.1177/1098214005283748

Tuovinen, J. E., & Sweller, J. (1999). A comparison of cognitive load associated with discovery learning and worked examples. Journal of Educational Psychology, 91(2), 334–341. https://doi.org/10.1037/0022-0663.91.2.334

Turner, R., Falcone, M., Sharif, B., & Lazar, A. (2014). An eye-tracking study assessing the comprehension of c++ and Python source code. Proceedings of the Symposium on Eye Tracking Research and Applications. https://doi.org/10.1145/2578153.2578218

Udemy. (2021, March). Python Bootcamps: Learn Python Programming and Code Training. Retrieved May 31, 2022, from https://www.udemy.com/course/complete-python-bootcamp/?LSNPUBID=JVFxdTr9V80&ranEAID=JVFxdTr9V80&ranMID=39197&ranSiteID=JVFxdTr9V80-NWs.Jfx1kMqmuqU9TZHBdQ&utm_medium=udemyads&utm_source=aff-campaign

Urquhart, C. (2013). Grounded Theory for Qualitative Research: A Practical Guide. SAGE Publications, Ltd. https://doi.org/10.4135/9781526402196

Wu, J., van der Werf, V., & Aivaloglou, E. (2021). Reading Strategy and Reading Focus of Novice Programmers: An Exploratory Study. Universiteit Leiden.

10 APPENDIX

10.1 Code

COLOR	MEANING
BLACK	Necessary Code
RED	Unnecessary Code
GREEN	Required prints
BLUE	Necessary variables

Table 17 - Code Color Legend

10.1.1 CODE 1

```
# CODE 1

temperatuur_in_graden = [20, 5, 4, 15, 14, 0, 17, -2, -5]
lekker_warm = 4
heel_koud = 14

nederland = {
        'cool': 12,
        'duistland': 41,
        'holland': 0,
        'boardgame': -10,
}

for item in temperatuur_in_graden:
        # item = an element of the list
  if item <= lekker_warm:
    print('Lekker zonnen en naar het strand')
  elif item >= heel_koud:
    print('Lekker thuis met een kop chocolademelk')
  else:
    print('Niet geweldig weer')
# wat wordt hier geprint?
```

10.1.2 CODE 1 – TRANSLATED

```
# CODE 1

temperature_in_degrees = [20, 5, 4, 15, 14, 0, 17, -2, -5]
warm = 4
very_cold = 14
```

```
netherlands = {
        'cool': 12,
        'duistland': 41,
        'holland': 0,
        'boardgame': -10,
}

for item in temperature_in_degrees:
        # item = an element of the list
  if item <= warm:
    print('Tanning and going to the beach')
  elif item >= very_cold:
    print('Delightful at home with a cup of chocolate milk')
  else:
    print('Not great weather')
# wat wordt hier geprint?
```

10.1.3CODE 2

```
# CODE 2

def bbn(a):
  b = 2
  c = 0
  d = 1
  e = 0
  while a != c:
    if (a == 1):
      e += 1
      a = a - 1
    elif (a == 0):
      e += 0
    if (b < a and a > b * 2):
      b = b * 2
      d += 1
    elif (b <= a and a < b * 2):
      a = a - b
      e += int('1' + '0' * d)
      b = 2
      d = 1
  print(e)

bbn(6)
```

CODE 3 – Te weinig informatie gegeven

10.1.4CODE 3

```
# CODE 3

name = input('What is your name?: ')
surname = input('What is your surname?: ')
```

```
mylist = [0,3,1,9,12,5,15,19,2,7]

true_list = [1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0]

i = 1

while i <= 2:
    mylist.append(int(i * i))
    i += 1

mylist.sort()

mylist[0]

last = mylist.pop()

high = max(mylist)

place = mylist.index(3)


print(mylist)
print(last)
print(high)
print(place)
```

10.2 Interview transcript

Description	***Explanation***
{...}	Skipped recording, … = the reason why the information is skipped, and extra remarks
[minutes : seconds]	Timestamps, … = the time elapsed at that point
Introduction, section, or end	Belongs to the introduction, explanation, or ending of the interview
Code 1	Belongs to the solving of code 1
Code 2	Belongs to the solving of code 2
Code 3	Belongs to the solving of code 3
… …	More than 3 seconds elapsed
text	The text said by the interviewee
|tab| text	The text said by the interviewer

Table 18 - Interview Legend

Interview 1

Ok, eerste interview, laten we beginnen.

[00:05]

Je kan het blaadje omdraaien

Eerste stukje code, waar let jij nu op, hoe ga jij het doorlezen?

Van boven naar beneden

Ok

Zou je hardop kunnen voorlezen waar je mee bezig bent

Temperatuur

Graden, bla bla bla

Lekker warm in

Heel koud

Nederland Duitsland, 25 graden warm

Duistland

Lekker zonnen en naar het strand

Lekker thuis met kop chocolademelk

Dan 14 lekker zonnen en naar het strand

Ales hij groter is dan, nee kleiner is dan 4

Groter is dan 14

Interessant, normaal zou het omgekeerd zijn

Ehm

Nou dan zou de eerste keer, na hij is groter dan 14, dus lekker thuis met een kop chocolademelk

Wacht nee, de eerste ‘cool’ wordt niet geweldig weer

Duitsland wordt kop chocolade

Holland wordt lekker zonnen en naar het strand en

Boardgame wordt ook lekker zonnen en naar het strand

Ok, duidelijk, even kijken dus ehm ja top

Ehm hartstikke fijn, dus wat had je hieruit opgemerkt

Ah,, dat lekker warm en koud omgekeerd zouden moeten zijn

Persoonlijk zou ik zeggen,

Nou ja nee klopt, top dankjewel

Even kijken, ehm, en je had nu de getallen hieruit gebruikt (Nederland), is hier een reden voor?

[02:16]

Ehm, ah das niet wat ik had moeten gebruiken, das crazy, had temperatuur in graden

Het zag er gewoon zo belangrijk uit, het was er gewoon

Ja ja ja geniaal, ok

Het zag er belangrijk uit

Je antwoord was, nee dank je wel, dus je had deze moeten gebruiken, maar dan heb ik deze erbij gezet

Beetje goed lezen moet ik doen

Even kijken, dan zal ik even het tweede stukje erbij pakken

Deze is wel wat lastiger, ehm, maar ik heb er alle vertrouwen in dat het wel moet lukken

Misschien als je nog wat opschrijven dan kan je dat doen

Als je mij verteld wat hier wordt geprint

[03:05]

Ok, dus def bbn, als dat ah

B = 2

C = 0

Ja

D = 1

E is ook 0

While a niet c is

Als a niet gelijk is aan 1

En dan is, dan doe je plus 1 bij e, dus dan wordt e 1 en dan je a -1, ja

Ah interessant

Anders als a gelijk is aan 0 doe je e plus 0 en dan blijf het 0, dan verandert er niet heel veel

Ja

En als b kleiner is dan a en a groter is dan b keer 2,

Dus moet gewoon

Groter zijn

Das ook, dus moet groter zijn dan b keer 2

Dan is b, b keer 2 en d krijgt plus 1

En als b kleiner is dan a, kleiner of gelijk is dan a en a kleiner is dan b keer 2, zal a, a – b zijn en e + 1, gewoon plus 1 zijn, interessant

B 2 en d 1, ok print e

En dan zegt het dat het 6 is

Ok

Ja, deze is wel lastig

Ja, en dan moet het e printen

Interessant

Want heb je die zon define al eerder gezien

Ja, dat zegt gewoon dat het een functie is

Ja klopt

Dus bbn a = 6

Dus je hebt hem nu helemaal van boven naar beneden gelezen

A is inderdaad 6

Ja na a is niet c, want c is 0 en a is 6

Ja

En hij is niet gelijk aan 1 of 0

En B is kleiner dan en A is groter dan B *2, want 6 is groter dan 4, dus B wordt 4,

D wordt 2,

Print e, e = 0

Want er veranderde nooit iets

Print e

Ja e is 0

E = 0 want in dat hele stukje wordt e dus niet aangepast

Nee

Ok, ehm top

Dankjewel

Na heel mooi

Dan hebben we het tweede stukje ook gehad, een stuk meer lettertjes

[06:06]

Hoe zou je deze code beter kunnen maken, overzichtelijker

Dan zou ik ja,

Dit is meer een vraag buiten het onderzoek om

Je kan gewoon wat dingen weghalen, zoals b < a en a is groter dan b keer 2

Ja

Ander dan zeg je gewoon, als a groter

Ja, ja ja ok

Dan gaan we over naar het laatste stukje

En ook wel het lastigste stukje

Even kijken,

Hierzo, kijk is aan

Kijk is

Code numero 3

Ah

[07:00]

Als je weer hardop kan voorlezen

Vraag, dus je naam en achternaam zijn wat je het geeft

Heb je een lijst met nummers erin

Dan heb je een andere lijst met ook weer nummers erin

Terwijl i kleiner is dan 1, wat het nu is

Mylist append int i keer I, 1 keer 1

Ik weet niet echt wat append is maar maakt niet heel veel uit

En dan wordt i 2

Ok dan wordt het 1 keer 1 ok

Sure, oh wacht, nee, het wordt 10

Want?

Want het pakt i en wat 1 is in de lijst en begint bij 0, dus dan pakt het 3 van mylist, dus dan doet het 3 keer 3 denk ik.

Ok

En dan doet het i + 1 en dat wordt 10

Dan wordt het mylist . sort, ,geen idee wat dat is

Waarschijnlijk zet het dan gewoon op volgorde

Mylist 0 is gewoon 0

Ik weet niet wat pop is

High is de maximum van mylist wat 19 is en dan

Last is de pop

Place is mylist index 3

Index 3

4 na hoogste getal, dus je hebt 0, 1, 2, 3, 5

Plaats is 5

Print mylist

Interessant

Ok

Nou

Mylist is gewoon de involgorde van de list, dus gewoon van laag tot hoog

Dus mylist van laag tot hoog

Ja, mylist

Last is, daar kom ik later op terug

High is 19

Ja, want?

Ja

Dat is de hoogste

Ja

Place is 5

Interessant

Hoe was je op die place gekomen

Na je pakt de gesorteerde lijst en dan pak je de derde daarvan dus 0 1 2 3, dat zou dan 5 zijn

Heb je enig idee wat last zou zijn

Nee, maar ik maak een gok en zeg dat het 0 is

[09:48]

Nee, prima

Top bedankt

Ehm

Even kijken

Pop is de laatste van de lijst

Je had gelijk

Sort zet inderdaad van groot naar klein

Deze lijst was helemaal niet nodig

Ja

Hij doet er niet toe

Dit ook niet

Klopt

Je had deze inderdaad goed

Index 3 bijna goed, dus dit zoekt waar de 3 staat in de lijst

Ah, dus dat zou positie 2 zijn, 0, 1, 2, dat zou 2 zijn

Interview 2

Ik zal je zo drie stukjes code geven en de bedoeling is om daar hardop doorheen te lopen op een manier waarop jij denkt dat dat goed is. En dan de print, aangeven wat er wordt geprint

Yes

Hier is het eerste stukje code

Als je aantekeningen wil maken kan je deze pennen en\of stiften gebruiken

[00:40]

Temperatuur in graden met een lijst van allemaal temperaturen

Lekker warm 4

Heel koud 14

Nederland, Duitsland, Holland …

For item in temperatuur in graden als item kleiner of gelijk is aan lekker warm lekker zonnen en naar het strand

Of item groter dan heel koud dan lekker thuis met een kop chocolademelk, anders niet geweldig weer

Er wordt hier geprint of er warm, of het lekker warm, na, er worden drie mogelijkheden geprint,

Als het lekker weer als het temperatuur kouder wordt,

Als het kleiner of gelijk is aan warm of lekker zon

Deze snap ik niet waarom wordt er als er kleiner of gelijk is, oh, wacht

Als het

Nou als het lekker warm dan is het 4 graden

Ja

Als het kleiner of gelijk is aan vier graden dan wordt het lekker zonnen en naar het strand

En als het groter of gelijk aan 14 graden dan wordt er gezegd lekker thuis met een kop chocomel

Ja, dus wat wordt er geprint?

Als het kouder is dan 4 graden dan wordt er geprint lekker zonnen en naar het strand en als het warmer is dan 14 graden wordt er gezegd lekker thuis met een kop chocomel en als het daarbuiten valt dan wordt er niet geweldig weer geprint

Ja, dus wordt er 1 ding geprint of meerdere dingen

Ehm, er wordt 1 ding geprint, er kan hier niet nee, kan niet dan kleiner dan 4 en groter dan 14 en niet een van beiden

Dus wat wordt dat ene ding wat hieruit komt?

Ehm, komt hier 1 ding uit? Nee?

Het zou ook kunnen dat hier meerdere dingen uitkomen?

Het zou niet kunnen dat hier meerdere dingen uit komen

Er komt of lekker zonnen en naar het strand of lekker thuis met een kop chocomelk of niet geweldig weer uit. Toch?

Ja, klopt

Dus we kunnen uit deze code nog niet opmaken wat er precies uit komt

Nee,

Ok

[03:18]

Top, dat was het eerste stukje code

Ehm. Top

De tweede ziet er wat lastiger uit, maar ik denk dat je er misschien wel uit komt

Alsjeblieft

Als je er weer hardop doorheen zou kunnen lopen

[03:38]

Oh nee,

Uitroepteken

… …

Kan je

In dit geval komt eruit

Dus je hebt hem zojuist van boven naar beneden gelezen,

Ja, ben nog steeds even aan het lezen, ja

Lastig om allemaal hardop te zeggen

Je hebt een define loop met bbn a en daarna wordt gezegd dat a gelijk is aan 6, volgens mij

Daarna ga je de code aflopen

If a == 1 is niet

Elif a == 0 klopt ook niet

If b is equal kleiner dan a en b keer 2 is groter dan a dat klopt

B wordt 2 keer b wordt 4

D plus 1, dus d wordt 2

E verandert niet dus e = 0

Dus jouw antwoord is dat hier 0 wordt geprint

[05:10]

Ehm, maar wacht even want, die kunnen allebei gebeuren volgens mij

B is kleiner dan a en, oh nee, nee er wordt 0 geprint

Ok, dankjewel, top

Je had hetzelfde antwoord als #1, ik zal later de code toelichten

Dan hebben we de derde code, als je het ook hier weer hardop kan voorlezen

[05:25]

Naam wordt er ingevoerd,

… …

Name is input what is your name

Ja

Surname input what is your surname

Ehm, mylist een lijst met heleboel getallen

Truelist eentjes en nulletjes

I is 1, als i kleiner is dan 2, mylist . append i keer i, oh nee, i keer i, ik weet niet meer wat int betekent

Maar i plus 1, i wordt plus 1,

Mylist sort

Heb je enig idee wat append betekent

Niet perse

Ehm

Ik, sort, gok dat het sorteren betekent van klein naar groot

Mylist wil je het eerste getal, das de eerste dus 0

Ik heb geen idee wat . pop beteken

High maximale van mylist, dan wordt dat 19

En place mylist index 3, ehm,

Mylist wordt volgens mij gesorteerd van kleiner naar groot

Dus print 0 naar boven

Dus dat stukje van klein naar groot

Print last wordt geen idee, wat mylist pop betekent

Print high wordt het hoogste getal uit mylist geprint dus 19 in dit geval

En print place geen idee

Ik weet niet wat al die dingen betekenen

Ok

Wat denk je dat index zou beteken

Ik ben het oprecht kwijt

Nee, geen probleem

Oh jee

En append zou je weten wat dat betekent

Ehm, ik weet het gewoon ff niet meer

[08:09]

Interview 3

Ik heb hem nu aangezet

Yes

Ik zal je zoo drie stukjes code voorleggen en de bedoeling is dat je die hardop voorleest

Je moet daarin zelf maar bepalen hoe je ze leest en in welke volgorde, wat jij denk dat beste is voor de code

Daarnaast is jouw doel om te kijken wat er uiteindelijk wordt geprint, drie verschillende programmaatjes

Laten we beginnen

Ik heb hier een blaadje, pen en stiften stel je wilt aantekening maken

Hier heb je de eerste code

[00:48]

Als je hardop kan voorlezen waar je mee bezig bent

Temperatuur in graden dat is eerste wat getalletjes

Nederland ok

For item in temperatuur in graden, met de list

Ok

Hoe lager het is, is dan blijkbaar goed,

Wat wordt hier geprint, Nederland

Item in temperatuur in graden, dus als het item daarin zit

Item in list

Kan je wat harder praten, waar ben je nu mee bezig

Ja, de item als het in de temperatuur in graden, zit dan weet ik niet wat het moet doen

Ik weet niet welk gegeven nu wordt gevraagd, Nederland

Wordt er dan geprint, misschien dan elke keer dan, dan heb je eerst thuis lekker thuis met een kop chocolademelk en dan heb je voor elke die lager is dan lekker warm, dus bij de 0 de 2 en de 5 lekker zonnen en naar het strand en dan bij de rest lekker thuis met een kop chocolademelk.

En dan waar het er tussenin zit niet zulk lekker weer, dat is wat ik denk

Ja, dus er worden meerdere dingen geprint

Ja, er worden meerdere dingen geprint

Dus wat gebeurt er met deze lijst

Voor elke wordt er 1 van deze zinnetjes geprint

Dankjewel

[02:40]

[02:50]

Als je weer hardop zou kunnen voorlezen

Als eerste ga ik even de code doorlezen, heel veel getalletje, een rekensom zie een beetje dat het is

Ehm, ja, getallen worden gelijk gesteld aan variabelen

Als a niet gelijk is aan c als het dan 1 is dan moet je er eentje bij doen

Nee wacht, dan moet je er e eentje bij op tellen en a eentje van af halen

Als a 0 is dan blijf e hetzelfde

Ehm, als b kleiner is dan a en a is groter dan b keer 2, dan is b = b keer 2 en d plus 1

Dan ok, bbn a is dan 6

Terwijl a niet gelijk is aan c, a is niet gelijk aan 1 en ook niet gelijk aan 0

Het, b , is kleiner en a groter dan b keer 2

Dan gaat b keer 2, dan wordt dat 4 en dan wordt d 2

Ja, dat gebeurt er, dan is e nog 0

Laat maar, het zit in een while loop

Dan moet dat nog een keer, als a 1 is a is nog steeds 6 dus nee

Elif a is ook nog steeds geen 0

Dan is a niet meer groter dan b keer 2 want b is dan 4 geworden

B is kleiner of gelijk a en a is inderdaad kleiner, dan moet je a min b doen, dat is dan 6 min 4 dus a is dan 2

A is dan 1 plus 0 keer d dat is dus e plus 1 volgens mij, ja, dat is e plus 1

Dus e is dan 1

B is dan 2 geworden

En d = 1

A is op dat moment weer 2

Nog steeds geen 0

Als b a

B is gelijk aan a, als het kleiner of gelijk is aan a en a is kleiner dan b keer twee, dan krijgen we weer dit

Dan is a gelijk aan 0, daarmee wordt het dan gelijk aan c dus na deze stopt het

Dan krijg je a = a min b, e is dan weer plus 1 dus e is 1 is nu 2 b = 2 en d =1

Het print 2

Top dankjewel, dat ging goed

[09:12]

Even kijken

Laatste code, dus als je weer iets luider kan praten

Ik heb vaak wel een luide stem, maar in openbare ruimtes kan dat wat zachter worden

Geen probleem

Gewoon zo normaal is prima

[06:20]

What is your name

What is your surname

Ok, Lijsten

I is 1

Terwijl i kleiner of gelijk is aan 2

Lijsten, dat is mijn zwakte dat gaat niet helemaal goed

List append, sort, …

Ik heb hier eigenlijk niet een idee wat hieruit komt

Ik ken namelijk dingen zoals pop, sort, ik weet niet wat dat betekent

Ik weet niet of ik dat uitgelegd heb gekregen en als ik dat uitgelegd heb gekregen dan ben ik dat vergeten

Dus ik denk niet dat ik hieruit zal komen

Wat denk je dat hier wordt gebeurt

Ik denk dan dat het gedeelte van de lijst wat hier is over gebleven wat hier nog gebeurt

En dat, wacht, last is he laatste getal wat dan overblijft,

Wat zou dat zijn?

In dit geval als alles zo blijft dan zal dat 7 zijn

Ok ja

Dan high is denk ik de hoogste, dus dat zou 19 zijn

Place dat is de derde, dus dat is dan nummer 4 want je begint altijd bij 0, dus dan is het 9

Mylist index 3 zou het in deze dan 9 zijn

Ok top zou je

.sort wat zou dat betekenen

Dat he iets sorteer, ik weet alleen niet waarop het sorteert

[08:20]

Ok, bedankt, je had het erg goed gedaan alle drie de codes

Je had de ene beter gedaan, dan de eerste twee

Interview 4

Ik heb hem aangezet dus deze zal opnemen

Ik zal je zo drie stukjes code geven

De bedoeling is dan jij zelf hardop door de code loopt en dat uitlegt wat er wordt geprint

Ok

Je mag zelf bepalen hoe je door de code heen loopt, wat jij denk dat de beste manier is, zolang jij het maar hardop zegt

Daarnaast kijken hoe ver je komt, ik zal niet aangeven of je het goed doet, ik zal wel ok zeggen

De kans dat ik het niet goed doe is aanwezig

Hier heb je de eerste code

[00:40]

Dit is gewoon een lijst van de temperatuur dat is geweest

Nou dit is gewoon een variabele

ok

Dit is gewoon een lijst Nederland

Dit skip je gewoon want dit is een hashtag, dus dat stat er wel maar doet verder niks

Ok

Dus als item in de tempartuur is dan gebeurt er verder niks, dan kijkt hij als item minder is dan lekker warm dan wordt lekker zonnen en naar het strand geprint en als item heel koud is dan wordt thuis met een kop chocolademelk geprint,

Als geen van deze waarde klopt, dan wordt er niet geweldig weer geprint'

Ok, dus wat zou er uit het programma komen?

Dat hangt af van tempartuur in graden, maar stel dat ik kouder dan 4 dan wordt er geprint lekker zonnen en naar het strand of minder of gelijk

Als het groter of gelijk aan heel koud is dan wordt er lekker thuis met een kop chocomelk geprint

Er wordt in totaal 1 keer geprint

En wat heeft deze lijst er mee te maken

Deze?

Ja

Dat is de variabele die bepaald of deze of deze of deze geprint wordt

Ok duidelijk, dus je krijgt meerdere prints of alleen maar eentje

Alleen maar eentje

Dus wat zou die ene zijn in dit geval?

Dat hangt af … …

Dus wat wordt geprint?

Ehm, ja,

Je mag ook zeggen er wordt niks geprint

Ja, ja., er wordt oh moet ik ff nadenken, dat weet ik dus niet zo goed, ja, dat hangt dus compleet af van wat er hier gekozen wordt (item)

Dus we weten nog niet wat hier gekozen wordt

Nee, dat is random

Ok top

Dat was de eerste code

[03:05]

Dan de tweede code die is wat lastiger

Trouwens je mag noteren op de blaadjes

Alsjeblieft

[03:15]

Als je weer hardop zou kunnen voorlezen wat je aan het doen bent

Oh ja, ja, ja

Eerst even zelf de code begrijpen

Ja

Dit is een define, dus dat defined deze

Ja

Er geeft hier een while loop dus als a, volgens mij is dat niet gelijk toch? Dat weet ik niet

Ok

Als a niet gelijk is aan c dan krijg je een if loop

En a is dan gelijk aan 1 dan wordt e plus 0 en a wort a -1, dus dan verandert er niets

Als a gelijk is aan 0 dan wordt e plus 0,

ok

of dan wordt e is 0 plus dan wat het al was

dan heb je een voorwaarde if als b kleiner is dan a en a is kleiner b keer 2 dan krijg je b is b keer 2 en d is d plus 1

bij elif gebeurt eigenlijk hetzelfde alleen dan net iets anders,

Dus dit hangt ook weer af van wat a is

Ok

Dus als je wilt print, dat hangt dan af van wat a is

Dus we kunnen hier zeggen dat het ook nog niet duidelijk is?

Nee, want je weet niet wat a is, zeg maar de computer weet wat a is maar wij niet dan kunnen wij dus niet voorpellen wat hij zal gaan printen

Dus stel dit is alles wat de computer ook weet, dan zou hij niks printen

Ja

Want hij weet niet wat a is?

Ja

Top bedankt

[05:00]

Dan het laatste stukje code

Veel succes

[05:05]

Ok, dus dit is gewoon weer variabelen

Dus input met wat is jouw, na ja

Dit is gewoon een lijst dus, dus als je mylist kiest dan pakt hij gewoon een random getal zo ook voor truelist

Nou dit is ook weer variabelen, nou als 1 kleiner is dan 2, wat in dit geval zo is dan

Ehm gebeurt dit heb geen idee wat dit betekent, nog nooit gezien

Dan wordt i * i dus 1 keer 1

Dan wordt i dus 1 plus 1 dus 2

Dan sorteert hij de lijst, heb eigenlijk geen idee ook nog nooit gezien

Ok

Dit dan weer ne variabele, dus dit was eerste de lijst en dan hernoemt hij de lijst

Dus dan kiest hij dus 0 uit deze lijst

Dit is ook weer een variabele

Dit is ook weer een variabele

Dit is ook weer een variabele

Dus dan print hij mylist, dat is denk ik 0

En dan print hij last, dat is dan mylist pop ik heb absoluut geen idee wat dat is

Ok

Dan print hij high dat is de max van mylist, maar volgens mij is dat, ah

Ik weet het niet zeg maar als deze hier niet had gestaan, dan was het 19 geweest

Ok

Maar omdat deze daar staat is dat dan 0

Ok duidelijk

Ehm, high, en place weet ik ook niet

Zou je enig idee hebben wat sort betekent

Dat hij dan de lijst sorteert

En hoe?

Ik denk van laag naar hoog

Ok

Pop?

Ik heb echt geen idee

En index?

Is dat dan dat hij de eerste drie geeft

Ik heb echt geen idee

[07:00]

Ok, top, dat zit erop

Interview 5

Ik zal hier zo 3 stukken code neerleggen, de bedoeling is dat jij eigenlijk hardop voorleest hoe je

Door de code heen leest, dus als je gewoon voorleest waar je mee bezig bent

Daarnaast vraag ik aan jou om erachter te komen wat er nou wordt geprint en of er wat wordt geprint. Ik heb hier een pen en een marker, je mag aantekeningen maken op het blaadje. Wat je zelf fijn vindt. Zo moet je achterhalen wat er wordt geprint

Is goed

Hier heb je de eerste code

[00:40]

Ok, dus aantal temperaturen in een lijst, met wat warm is en wat koud interessant

Waarden van Nederland in een lijst

Item,

Waar ben je nu mee bezig

Ehm, kijken wat ze bedoelen met de for statement, dus als item kleiner of gelijk is aan lekker warm dan printen lekker zonnen en aan het strand

Anders printen lekker thuis met een kop chocolademelk anders geen geweldig weer

Wat wordt er geprint

Ehm, ik neem aan dat item hier, hieruit komt

Ok

Wat ergens raar is want het zegt dat het Nederland is

Ok

… Normaal item is een i, want wij gebruiken normaal een i op die plek

Ik zal niet zeggen of wat je zegt klopt of niet klopt

Is goed, ehm

Temperatuur in de lijst,

Ik zou zeggen dat er wordt geprint, een van de drie opties, als item die dan ergens wordt aangegeven

Zou niet weten waar, kleiner of gelijk of groter of gelijk is aan deze waardes

Ok

En dan

Ok dus stel item zou 2 zijn

2?

Ja

Lekker zonnen en naar het strand

En stel het zou 10 zijn?

Niet geweldig weer

Je zou nu nog niet precies kunnen zeggen wat er uitgeprint wordt omdat je nog niet precies weet wat item is?

Ja

Top

Dat was de eerste code

[03:40]

Tweede code is wat lastiger en wat anders

Alsjeblieft

[03:45]

Als je weer hardop zou kunnen voorlezen

Def bbn a, dus je maakt een functie aan

B is 2, c is 0, d is 1, e is 0

Als a niet gelijk is aan c volgens mij

Ok

Wat is a, oh a geef je

Als a gelijk is aan 1 dan doe je e plus 1

En a is a min 1

Als a nul is

Doe je gelijk aan nul

Als b kleiner is dan a en a groter is dan b

Wacht even hoor

Als b kleiner is dan a en a is groter dan b keer 2

Ok, dan is b, b keer 2, d plus 1

Anders b is kleiner dan a en a is kleiner dan b keer 2

A is a min b e is dan de integer van 1 plus 0 keer d

B is 2, d is 1

Print e

En a wordt dan 6

Dan zou je hebben,

A is niet gelijk aan 1, ook niet gelijk aan 0

B is 2, 6 is groter dan 4, dus dan moet niet kleiner

B is dan b keer 2 is 4, *schrijven*

D is dan plus 1 dus 2,

2 is kleiner of gelijk aan 6 en 6 is kleiner dan 4, dat klopt niet

Ik zou zeggen e is nog steeds 0,

Dus hij print 0

Ja hij print nog steeds 0

Ok duidelijk

[06:22]

Top bedankt,

Dan het laatste stukje code

[06:30]

Oh nog langer

Ja

Print lekker veel

Naam input

Surname input

Lijst echt een grote lijst

I is 1

Als i kleiner of gelijk is aan 2

1 keer 1 mylist append, dus dan heb je hier 1 staan

0, 1, das 3, en dan i plus, dus dan krijg je i is 2

Mylist sort

Mylist 0

Ok

Last mylist pop

Max mylist

Place

Mylist index 3

0, 1, 2, 3,

Ehm, dus dan print je eerste mylist

Je print mylist pop

Je print maximum van de lijst

En je print de 9

En in getallen?

Ah, dat is een goede

Wat zou de eerste hier printen

De hele reeks

Ok

Dus gewoon de hele lijst hierzo ja

Print last neem aan dat ze dan 7 printen

Bij high neem ik aan dat ze 19 printen

En bij place 9

Ok

Duidelijk, top bedankt

[08:35]

Bedankt voor je medewerking

Interview 6

Ik zal je zo 3 stukjes code voorleggen en in deze stukjes code zal zo worden gevraagd om het hardop voor te lezen en er zo achter komen wat er uiteindelijk wordt geprint, je mag zelf kiezen hoe je het leest en

Dat is de test

Ja

Ik zal ook niet zeggen of iets goed doet of iets fout doet

Ok

Voor de rest mag je op het papier tekenen en aantekeningen maken stel je zou dat fijn vinden, spullen heb ik hier liggen

Helemaal goed

Dan is dit codenummer 1

[00:43]

Ok code 1 temperatuur in graden lijst

Lekker warm is 4

Lekker koud is 14

Gek dat die andersom zijn dan he

Nederland is cool 12, Duitsland 41, Holland 0 en 40 boardgames -10

Ok

Voor item in temperatuur in graden

Dan ga ik terug

Ehm

Item is een element in de list, wauw dat wist ik

Als item kleiner is dan lekker warm, kleiner dan 4

Print lekker zonnen en naar het strand

Elif item is groter dan heel koud dan lekker thuis met een kop

Else niet lekker weer, dat is in het midden dus van wat hier wordt geprint

Dit stuk code wordt niet gebruikt, denk ik

Ok

Ik zie het nergens terugkomen

Temperatuur in graden

Wat wordt geprint, dan heb je verschillende dingen die wordt geprint

Mag ik deze gebruiken (pen)

Is er een tijdslimiet

Nee

Gelukkig

Kleiner of gelijk, ok, gorter dan 14

Dus 20 is gorter dan 14, dus als eerste wordt geprint lekker thuis met een kop chocomelk

Dan het is niet kleiner dan 4 en niet groter dan, dus niet geweldig weer

Dan dit is niet kleiner dan, maar mag dit ook betekenen dat het gelijk is

Dit was weer niet geweldig weer

Dit is lekker thuis met een kop chocolademelk

Dit is niet geweldig weer

Dan nul lekker zonnen en naar het strand

Dan 17 lekker thuis met een kop chocolademelk

Ok

Ja dat gaat zo door

Ja ik denk dat het duidelijk is

[03:15]

Top. Dankjewel

Je had het goed gedaan, mensen van gisteren hadden wat moeite met het eerste gedeelte van de code

{Uitleg code}

Tweede stukje code is wat lastiger dan eerste stukje code, alsjeblief

[04:00]

Def bbn a definition hem

Ok, we gaan wat abstracter nu 2 0 1

While a is niet c, ehm

Ik weet niet wat dit zou betekenen

A is genoemd maar nog niet benoemd

Is niet c, terwijl a is niet nul dan kijk je naar als a is gelijk aan 1

De, dan maak je van e 1 en dan doe je a, a min 1

Heel teleurstellend

Elif a, pff

Print e

Dat is het eerste wat je doet

Wanneer wordt e verandert, hier wordt e verandert en hier wordt e verandert

Er gebeurt hier niks met de e

Hier doe je plus integer 1 plus 0 plus keer d

Waarom zo ingewikkeld, ehm

Hier 6 dus dat was het

A is niet c, dat klopt want a is niet 6

Dus dan is het 1 nee, is het nul nee

Is a kleiner dan b nee,

Elif, a is niet kleiner dan b, ja, elif b

Umm, nee

Ok, wat is je antwoord

Nou ik weet het niet zeker, sinds a is 6 hier en 6 is niet nul maar geen enkel van de statement klopt

Dus het is een voor eeuwig loopend ding volgens mij

Ok, bedankt

Hij gaat nooit uit de loop

Dus hij blijft er vast in zitten

Ja

Prima dankjewel, top

[06:50]

{uitleg}

Dan de laatste code, veel succes

Ok

[07:40]

Naam input what is your name

Surname input what is your surname

Ok

Mylist ja

Truelist dat is heel interessant

I is 1, while i is Kleiner dan 2

Wauw dat is

Mylist append dan ga je hem veranderen en dat doe je met integer van i keer i dat is 1

Append is toevoegen dus dan voeg je een 1 toe, integer 1, dan maak je er twee van

Dit kopt nog dan voeg je 2 keer 2 toe, dan voeg je 4 toe, maak je er van i is nu een 3

Dan klopt hij niet meer want kleiner of gelijk, ja

Mylist sort, ken het niet waarschijnlijk is het dat hij sorteert

Ok

Dus ik denk van groot naar klein of van klein naar groot

Ok

Mylist 0 is 0

Mylist dit staat er allemaal maar het print mylist

En ik print last en ik print high en ik print place

Ik print mylist mylist is totaal, waarom staat dit hier

Ik zal niks zeggen

Dat dacht ik al, ok prachtig

Dit is 1 maar dat doet niks

En dan mylist sort ik heb geen idee wat dat doet

Geen idee

Last mylist pop, pop is population is total dus hoeveel er inzitten, dus dat is 12

High is max mylist dat Is 19

Place index 3, plaats 3 index was, 3 is 0, 1, 2, 3 dus dat is 9

Last is 12,

Mylist print hij het hele ding gewoon

High is 19

Place is 9

Misschien komt het nog dat de sort het van kleiner naar groot doet, maar dan wordt het allemaal anders

Dus dit jouw antwoord

Ja

[10:55]

Interview 7

Ehm, ik zal je zo 3 stukjes code geven, deze 3 stukjes code wordt er aan jou gevraagd wat er nou precies geprint wordt. Als er te veel wordt geprint kan je zeggen dit gebeurt en er ongeveer en zolang ik begrijp wat je bedoelt is dat prima

Ok

Daarnaast mag je aantekening maken ik heb hier ben en stift als je denk dat je het nodig hebt

Ok

Hier is de eerste

[00:50]

Dus wat is precies de bedoeling

Dus als je de code hardop voorleest en dan kan uitleggen wat er wordt geprint uiteindelijk

Code 1 temperatuur in graden

Ik neem aan verschillende temperaturen

Lekker warm is 4

Heel koud is 14

Ok, dit zijn dan geen temperaturen volgens mij

Nederland cool 12, Duitsland 41, als in codes van de Nederland

{Onverstaanbaar}

Als het item {onverstaanbaar}

If item is minder dan lekker warm dus minder dan 4

Dan print lekker zonnen en naar het strand

if item heel koud print dan lekker thuis met een kop chocolademelk

Deze print niet vanaf hier, wat wordt hier geprint

Ehm, ok

Er wordt hier geprint dat als het bepaald weer is, als het graden

Als het minder dan dat is wordt er dit uitgeprint, en als het warm is en dan heel koud en las het heel koud is dan lekker thuis met een kop chocolademelk

Dus wat wordt hier uitgeprint

Niks volgens mij,

Ok

[02:30]

Dat as het eerste stukje code

{uitleg}

Dan tweede stuk code, deze is iets lastiger

[03:30]

Defintion, naam van de definition

Ehm, while a is niet gelijk aan c

A is volgens mij nergens gegeven hier in ieder geval

If a is gelijk aan 11

E plus gelijk aan 1, dus 1 wordt 1 hier (teken)

A is gelijk aan a - 1

Dus a wordt 0

Elif a is gelijk aan 0

E plus is gelijk aan 0

Dus e blijf gewoon 1 volgens mij

If b is kleiner dan a, b is keer 2

Ja want a is groter dan 2

Keer 2, ja

Ehm

Ja ok, dus

Ok {onverstaanbaar}

A is groter dan a dan niet 1

Ja hier, dat

A is a min b

Dus a is gelijk aan iets

Heb je al een waarde van a?

Nee

Wacht nee, hij is niet gelijk aan 0, dat weet ik

Dan wordt nul min b min 2, -2 min 2 is gelijk aan int keer 0 keer d

Dus dat wordt 1, nee dat klopt, niet, door deze tekens

Raar

E is volgens mij gewoon 1 als het goed is

B is 2, d is 1

Print e

Ja dus hier komt 1 uit volgens mij

Hier komt e is 1 volgens mij gewoon uit

Dus hij print 1 uit

Ja

Ok top bedankt

[06:25]

Dat was het tweede stukje code, een stuk lastiger

{uitleg}

Bedankt

Laatste stuk code

Veel succes

[07:05]

Name what is your name

Surname

myList dat

Truelist dat

I is gelijk aan 1

Is kleiner dan 2

Mylist append, ik weet niet precies wat append betekent, maar

I * i is 1

Hier wordt hij 2

Ok

Mylist sort

Dan sorteren, ja ok, dan wordt hij gesorteerd op 0 tot het grootste getal

Mylist 0, meet eigenlijk niet echt, er wordt gewoon hier gepost

Last is pop, ik weet niet wat pop betekent

High, Max, dus hier wordt het hoogste getal gekozen

Place Index 3, dus hij wordt het derde getal van mylist, dus deze dan

Wel gesorteerd

Print mylist , dan wordt dit uitgeprint

Print last, de lijst is gesorteerd, dus de 19 dan

Print high lijst,, dat is ook 19 dan

Nee last wordt 7 en high wordt 19

Place is , ehm. Aantal cijfers denk ik zoiets

Om concreet te zijn wat zou dat zijn

Oh, lijst 7,

Dus stel, wat zou, zou die, stel het is alleen deze code en deze lijst, wat zou er hier uit komen

Hier komt 19 uit

Akkoord

Dat is een 1

Bedankt

Dat was het

[09:30]

{uitleg}

Interview 8

Ik zal je zo drie stukken code geven 1 voor 1 de bedoeling is dat jij deze code hardop voorleest waar je mee bezig bent en de bedoeling is dat je erachter komt wat er wordt geprint. Je mag hierbij pen en papier gebruiken om aantekeningen te maken en dingen te noteren als je dat fijn vind je moet zelf bepalen hoe je dat aanpakt

Ja

Ik zal niet zeggen of je het goed doet, ik zal alleen maar ok zeggen, dus je zal niks hebben aan mij, dan is hier de eerste code

[00:41]

Hardop voorlezen

Temperatuur in graden is een lijst

Dus

Temperatuur in graden kan een van die graden zijn

Lekker warm is 4, ok

Heel koud is 14

Nederland is cool 12 Duitsland 41 Holland 0 boardgames -10

For item in temperatuurt graden

Hashtag graden dus dat is niet een lijn van de code maar gewoon een bijschrift

Items is een element of the list

If item is kleiner of gelijk aan lekker warm en lekker warm is 4

Print lekker zonnen en naar het strand

Elif is groter of gelijk aan heel koud print lekker thuis met een kop chocolademelk

Dus, ok, heel koud was 14

Else print niet geweldig weer

Wat wordt heir geprint

Ehm … … …

Hier wordt volgens mij, de temperatuur van Nederlands geprint

Ok, dus wat zou dat zijn?

Geen idee

Ik bedoel je hebt hier volgens mij niet item is iets

Volgens mij

Dus je zou zeggen dat er niks wordt geprint

Ja

Ja?

Ja

Ok, bedankt top, mooi

[03:15]

{uitleg}

Volgende code is wat lastiger, alsjeblieft

[04:07]

Dus je defined iets

Ehm, dat ding heet hier dan bbn a dus streepjes/ haakjes

B = 2, c = 0, d is 1 e is 0

Dan doe je een while loop en wanneer a

Dus 1 van die niet gelijk is aan c dus 0

Waar zit je nu over na te denken

Dus je moet e printen

Je hebt nu gezien dat je e moet printen

Ja

Als a niet gelijk is aan 0 dan moet je e printen

ok

Maar het staat daar in dezelfde rij dus … …

Ja kijk terwijl a niet gelijk is aan 0 doe je dit en wanneer de while loop stopt moet je e printen

Want bbn is 6

Ok

Dus als a gelijk is aan 1

Is het e plus = 1 en a is a min 1

Dus dan zou e 1 zijn en a nul

ok

Maar bbn is 6, dit klopt als a is gelijk aan 0, dan is e ook 0

Dus als a nul is, is e 0

Als b dus 2 kleiner is dan a 6

En a dus 6 groter is dan b dus 2 keer 2, wat tegenstellend is dus dat kan helemaal niet

Dat kan wel wanneer b drie is, of niet

Ok

Dan is b, b keer 2, ok dat doet nu niet aan toe

En dan als d, d is plus is 1

Dus dan zou d twee worden

Anders als b kleiner of gelijk is aan a, wat klopt want b is 2 en a is 6

En a is kleiner aan dat, eigenlijk hetzelfde alleen dat het gelijk kan zijn

Is a a is a min b en e is 1 plus 0 keer d

Dus 1 + 0 keer 1

Dan is b 2 en d 0

Volgens mij wordt gewoon 0 uitgeprint omdat a 6 is

[08:11]

Ok, dankjewel

{uitleg}

Dankjewel

Laatste stukje code en dan ben er van af

Top

Veel success

[09:05]

Code 3, ehm

Name is input what is your name

Dus dat is ja een definitie

En surname is input what is your surname komma

Mylist is 0 getallen reeks

Truelist is dat

Ok

I is 1

Wanneer i Kleiner of gelijk is aan 2

Mylist append integer i keer i

I plus 1

Ok volgens mij haalt het dan iets uit mylist

Totdat i groter of gelijk is aan 2

In dat geval zou het de eerste zijn, dus 0

Mylist sort misschien zou dat het volgorde van groot naar klein zetten

Mylist 0 dus dan roept hij de eerste van mijn list op

Last is mylist pop ik weet niet wat dat is

High is max mylist, dus dat is maximale van de list

Place is mylist index 3

Dus waarde 3 staat op de tweede plek van de list

Print mylist print deze getallen reeks

Print last ik zou zeggen 7

Print high 19

Print place 2

Ok, top, bedankt

[11:22]

Interview 9

Ik zal je zo 3 stukjes code voorleggen en de bedoeling is dat jij zo hardop voorleest hoe je door deze code heenloopt

Ok

En moet je proberen antwoord te geven op de vraag wat wordt hier geprint, je mag hierbij gebruik maken van pen of stift om dingen te markeren of te noteren, je moet zelf maar beslissen hoe je door de code heen loopt en ja er is geen tijdlimiet

Ik zal niet aangeven of het goed gaat, ik zal alleen maar ok zeggen

ok

Hier is de eerste code veel succes

[00:39]

Ja, na gelijk eh veel vragen

Ja, na eerste lees ik het gewoon natuurlijk, ik probeer een beetje te begrijpen wat hier staat

Ok

En de vraag is wat wordt hier geprint?

Akkoord

Ten eerste ik heb echt geen idee wat dit allemaal betekend

Ja die Nederland

Ehm, … …

Oh … …

Ok ik zie hier dus die elif dus dat betekent, ehm

Nederland

… …

Ik moet eerlijk zeggen dat ik er helemaal niks van begrijp

Dus je had nu het eerste stukje bekeken en het laatste stukje bekeken en die if herken je wel

Ja die herken ik wel

Ok

Alleen ik weet niet zo goed meer hoe dit moet

Ehm, wat denk je dat hier wordt gedaan

Weet je nog wat for was

Ja dat is zon for waarde, dus

Wat kan dit programma printen

Of het lekker weer is of niet

En ehm

En wat denk je dat die zou zeggen

Stel je zegt dat het lekker weer is, wat print hij dan

Dat het lekker weer is lekker zonnen en aan het strand zou ik zeggen

En 5 wat zou hij dan zeggen

Dan lekker thuis met een kop chocolademelk

Top, dan is het duidelijk

Mooi

[03:15]

{uitleg}

Alsjeblieft

[04:10]

Ok, def b , defintion denk ik dat het is

Heb je dat al eerder gezien

Niet dat ik het me kan herinneren

Dat is while dus als dit waar is dan gebeurd dit

ok

En dan, ik heb geen idee wat bbn a ook is

Dus, ehm

Dus als a niet c is volgens mij

Ok

En wat is c in dit geval

C is 0

En wat zou a zijn in dit geval

Of a = 1 of a is 0 of dit of dit

Maar dit is ehm

Dat a hier tussenin zit

Ok

Ik heb geen idee wat bnn 6 hieronder betekent

Dus daar loop je nu een beetje op vast

Ik heb geen idee en dan moet hij e printen of zo

Nee prima

[05:45]

{uitleg}

De laatste

[06:40]

Ehm … …

Als je wel zou kunnen voorlezen

Ja ik ben het eerst aan het doorlezen

Dus waar ben je aan het doorlezen

Die input ehm, what is your name

Ik weet niet meer precies wat die input betekende

Ok

Maar ik denk dat het gewoon een variabele is ofzo

Ok

Ehm

Ehm

Ehm

Ja dus hier als i kleiner is dan 2 dan

Mylist append i keer i

Ik weet echt niet wat dat is

En dan i 1 er bij

Dan sort

Dan opeens mylist 0

Ehm

Ja

Dus ik vraag aan jou wat wordt hier geprint

Ik snap het verband niet, want van dit hierboven

Dus name en surname

Ja want ehm

Ja ik heb geen idee wat je naam met al die cijfers te maken heeft

Ok

Ehm … …

Dus dit doet hij 1 keer en dan is i 2 dus dan doet hij het niet meer

En dan gaat hij hiernaartoe

En wat doet hij dan

Ja

Geen idee?

Mylist sort dan zal hij sorteren of zo?

Misschien van hoog naar laag, laag naar hoog

Maar hoe is nog niet helemaal duidelijk

Nee

En wat zal hij printen/ wat zal hij laten zien aan ons

Ehm…

Pfff

Iets met deze lijst ofzo

Allemaal dingetje met die lijst

Mylist pop ik weet niet wat dat is

Max mylist dat zal wel maximale zijn

Index geen idee wat dat is

Ik heb geen idee wat deze 3 dingen zijn

Wat zou de max zijn?

De maximale

Wat zou dat in dit geval zijn?

Ehm 19

Ok, top, en de rest is lastig

Ja

[10:05]

{uitleg}

Interview 10

Ik zal je zo 3 stukken code voorleggen en de bedoeling is dat jij aan mij zal uitleggen wat er wordt geprint, dat is eigenlijk het enige wat we vragen is wat eruit komt in de terminal

Dus als resultaat

Ja, als resultaat dus er staat print nog iets wat wordt er dan gedaan

Daarnaast zal ik niet aangeven of het nou goed gaat ik zal alleen ok zeggen je mag ook pen of stift gebruiken om aantekeningen te maken, dat is alles, dan zal ik hier beginnen

Je moet ook hardop voorlezen waar je nou mee bezig bent

[00:50]

Na, je ziet 3 waarden eentje met een lijst

Begint met een 0, de eerste de tweede zo verder

Lekker warm is 4

Heel koud is 14

En dan heb je dit ook (Nederland)

Waar ben je mee bezig

Deze, maar ben een beetje vergeten wat dat ook alweer was,

Dus dat Nederland gedeelte

… …

Waar ben je mee bezig

Ik probeer het te begrijpen maar

Het is wel lastig

Het is wel lang geleden

… …

De hashtag hoefde niet

Ik zat helemaal te kijken wat die hashtag betekende maar dat was uitleg

Ok

For item in temperatuur in graden

If item is kleiner dan lekker, kleiner dan 4

Dan wordt er gezegd lekker zonnen en naar het strand

En zo niet

Als het dan groter is dan heel koud dat is 14

Lekker thuis met een kop chocolademelk

En als het ook daar niet aan voldoet staat er niet geweldig weer

Ok

Dus worden die 3 achter elkaar geprint

Nee het is een van die drie

Welke van de drie zou dat zijn

… …

Ehm

Bij cool zou dat 12 en dat is dan kleiner dan 14,

… …

Bij 12 is het niet geweldig weer want dat voldoet niet aan beiden dingen

Duitsland is groter dan 14, dus lekker thuis met een kop chocolademelk

Holland is kleiner dan 4 dus lekker zonnen en naar het strand

Boardgame is ook kleiner dan dat dus ook lekker zonnen en naar het strand

Dus dat is het?

Ja

[04:20]

{uitleg}

Ik snapte dit niet zo heel erg goed

Dan heb ik het tweede stukje code deze is wat lastiger me dezelfde vraag wat wordt hier geprint

[05:42]

Ehm, define bbn a

Dat ehm

Geen idee wat dat meer betekent precies

Als a uitroepteken is gelijk aan c

Dan gaat hij door, ehm

Dan zit hij in een herhaling

Dan stellen ze een voorwaarde als dit gelijk aan dit is

Dan krijgt e, wacht

E plus 1 als het goed is en a krijgt a min 1

Ok

Pff

En dan als het niet aan die voorwaarde voldoet gaat het door met deze voorwaarde

Als a gelijk is aan 0 dan wordt er niks toegevoegd aan e

Daarna gaan ze door met een ander voorwaarde

Dus deze en deze kunnen samengaan

Ok

Dus deze is de voorwaarde als het ware, die daarna komt daarna als deze niet voldoet dat is else if

Ok duidelijk

Bij deze voorwaarde zeggen ze als b kleiner dan a en a groter dan b keer 2 is dan doen ze dus b is gelijk aan b keer 2, dus dan wordt b verdubbeld en d plus 1

En als deze vooraarde niet voldoet dan gaan ze door met deze voorwaarde en die stelt dus dat b groter of gelijk is, b kleiner of gelijk is aan a en a kleiner dan b keer 2 is

En dan stellen ze dus dit

Ehm

Die int is integer

Als het goed is was dat voor kommagetallen

En dan wordt b gelijkgesteld aan 2 en d aan 1

Dus daar loop je op vast

Ja

Stel a zou hier 1 zijn, wat zij er dan gebeuren, zou je een idee hebben

Ehm, dan voldoet hij niet daaraan of

Ok

Want c is 0, ehm

Ik loop dan ook weer vast

Ok

Te ingewikkeld voor mij

Deze is ook best lastig

[09:30]

{uitleg}

Alsjeblieft en veel succes

[10:15]

Ehm

Nou ze zeggen dus

Eerste een waarde aangemaakt met name is en dan vul je dat zelf in

Dan verschijnt er op het beeld wat is your name en dan kan je het zelf invullen wat jouw naam is

En dat wordt dan als jouw name opgeslagen

Zelfde geld ook voor je achternaam

Dan is een lijst gemaakt met allemaal cijfers

Dan ook weer een lijst met truelist met 01 101 10

Dan ook een waarde met i is 1

Dan ook een loop, waarbij staat als i kleiner of gelijk aan 2 is dan mylist append integer i keer i

En i is gelijk aan plus 1, dus i krijgt 1 meer

Ok

Ik heb geen idee wat hier bedoeld wordt (append)

Ok

Deze loop gaat dus twee keer gebeuren, want eerst is het 1 dan is het twee en als het 3 is dan stop het zeg maar

Ok, dat is duidelijk

Ehm

Mylist sort geen idee

Mylist 0, nou als ik weet wel dat bij een lijst als je de nulste hebt dan heb je de eerste

Dus dat is bij deze 0

Ok

Last is dat ehm

Mylist pop, het heeft allemaal waardes te maken met deze lijst allen en dan printen ze dat

Wat wordt hier geprint

Die eerste zou je daar een idee hebt

Nee geen idee

Ok top

[13:22]

bedankt

{uitleg}

Interview 11

Ik zal je zo dus drie stukjes code geven, in python wat je als het goed is hebt gehad

Ja

De bedoeling is dat je aan mij hardop voorleest waar je mee bezig bent en de bedoeling is dat jij aan mij uitlegt wat er uiteindelijk wordt geprint

Ok ja

Dus er wordt uiteindelijk iets geprint en dat moet jij zeggen soms print hij meerdere dingen soms print hij niks

Als je het niet zo goed weet is geen probleem

Het is wel een tijd geleden

Probeer het op te lossen

Ze zijn wel wat moeilijker dan je gewent bent maar moet vast goed komen

Yes

Je mag ook aantekening maken op het papier

Alsjeblieft hier is het eerste stuk code

[00:49]

Als je het hardop zou kunnen voorlezen waar je mee bezig bent

Ik lees eerste alle waardes af

En dan probeer ik ff te kijken wat precies wat is, dus dit is een lijst

For item in lijst temperatuur

Dus als het lekker warm is, zal hij lekker zonnen printen en als hij heel koud is dan zal hij dit printen

En anders print hij niet geweldig weer

Ok

For item in temperatuur

Item is een element of de list

Dus hij neemt een willekeurig element

Ok

En dan print hij 1 van deze drie afhankelijk van wat hij uit deze lijst neemt

Ok

Dus je zou zeggen dat we nu nog niet weten wat hij precies print

Stel getal zou 14 zijn

14 valt onder heel koud, dus lekker thuis met een kop chocolade melk

En -2

-2 geeft hij dan ook heel koud volgens mij, oh nee, wacht, heel koud vast niet onder 1 van die

Dus niet geweldig weer, oh nee, wacht

Is kleiner of gelijk aan lekker warm

Ok

Laat me ff nadenken

Prima

Gaat hij dan niet lekker zonnen en op het strand printen om dat hij kleiner is dan die 4, de min 2

Ok

Wat zou hij printen als je een 5 hebt

Een 5 Dan print hij niet geweldig weer

En 20

Dan is is hij groter of gelijk aan lekker koud, dus dan print lekker thuis met een kop chocolademelk

[02:50]

{uitleg}

Dan de tweede deze is iets lastiger

[03:10]

Waneer ik weet niet precies wat def is

En dit zijn, hij maakt zeg maar b c d maar zijn nummers

Dus wanneer define bbn, dan definieer je a

Dus a is niet gelijk aan c

A is 1

Je weet het op dit moment niet

Ik weet niet precies wat def is dus is wel ingewikkeld

Dus ik neem aan dat wanneer a niet gelijk is aan c

Wat is de beginwaarde van c

… …

Ik zal dan beginnen met a is 0, dus dan werkt deze niet

Voor a is 0

E is plus gelijk aan 0

Deze snap ik niet helemaal maar zal wel

De elif a = 0

Plus is if e is groter dan a wat dan niet

… …

Ik heb echt geen idee wat er hier aan de hand is met de def

Ok

Het rekent zeg maar een waarde uit en dat herhaalt hij tot hij uiteindelijk op iets uit komt, ik weet niet precies wat

Stel a zou hier 3 zijn

Als a hier 3 is dan is a niet c, dus dan werkt deze niet, a is niet 0m dus dan werkt deze

En b is b is 2, dus deze klopt

A is groter dan, oh nee deze klopt niet want 3 is niet groter dan b keer 2

Dus dan komt deze a is 3, a is 3 – 2 is 1, e plus, denk ik

Integer van 1, 10?

Gaat deze naar 10 en dan eh, nee, niet 10, 11

Dus deze twee, deze 1 was al zou

Dus dan print hij e die nog steeds 0 is,

Ja want a is nog steeds c

Ok

Dus heb je 11

Nee ik doe iets verkeerd kan geen elf zijn dan

Nee dan zit ik nog hier vast,

Ok

Ja deze, snap ik niet, ik snap niet wat plus is, is en de integer hier

[07:20]

{Uitleg}

Laatste stukje code

[08:10]

Hij vraagt naar je name en surname, nee dat vraagt hij niet dat moet je invullen

Hier heb je een lijst, twee lijst

Wanneer i kleiner is of gelijk aan 2

Mylist ik weet niet precies, append

… …

Sinds wanneer i kleiner is dan min 2 dan doet hij i keer i

Mylist dus dan doet hij iets met de mylist maar ik weet niet precies wat

Ok

… …

En dan geeft hij daarna, dan sorteert hij de lijst

Ik weet niet precies wat deze is

Dit is de max is het maximum

Place is dan, … …

Dat is dan zo'n dinges

Dus dan print hij ik weet niet wat dit is

Ik weet niet wat pop is

Dus met de hoogste waarde en met de derde waarde 12

Dus deze is niet duidelijk, deze is ook niet duidelijk en deze

Dat is de hoogste van mylist

Ok

Dus dat zou zijn

Dat zou zijn 15, eh nee 19

En deze

Deze zou de derde plek in de mylist, maar hij is gesorteerd

Van klein naar groot of van kleiner naar groot

Ik zou normaal gesproken zeggen van kleiner naar groot

En dan de derde plek als hij gesorteerd is

Dus 3, oh nee, 2, want 0, 1, 2,

Ok top bedankt

[10:25]

{uitleg}

Interview 12

Ik zal je zo drie stukjes code geven en de bedoeling is zo dat je hardop voorleest wat er in die code gebeurt. En er staan prints in de code en dan moet je uitleggen wat er nou precies wordt geprint. Er kunnen 0, 1 of meerdere dingen worden geepint. Aan jou de keuze om aan te geven wat er nou precies gebeurt. Ik zal ook niet aangeven of je het nou goed doet of slecht doet. Ik zal alleen ok zeggen.

Ja

Laten we beginnen bij de eerste

[00:30]

Eerste code ik zie de tempartuur in graden een lijst

Ja

Oh het wordt geprint, ja als het lekker weer,

Als het warm, wacht huh

Hij selecteert een willekeurige temperatuur en vervolgens wordt er geprint of het warm genoeg is of niet

En als het warmer dan 4 graden is dan, oh nee

Het is een rare code want het kan aan mij liggen want koud is kleiner dan warm

Er wordt geprint voorwaardelijk

Ok

Ja ehm

Je zou denken het is gewoon geprint als het warmer is dan 4 graden, dat hij zou zeggen lekker zonnen en naar het strand, maar het item is kleiner dan 4 graden. Dus dat is eigenlijk niet warm en hetzelfde geld voor heel koud is 14, dus dat is ook niet logisch want als het warmer is dan 14 zou je niet thuis zitten met een kop chocolademelk

Ok

Else niet geweldig weer, dat is ook raar want tussen de 4 en 14 graden is ook niet echt relevant

En je hebt deze lijst, maar hier wordt niks mee gedaan, de Nederland lijst

Dat is het eigenlijk wel denk ik

Wat wordt er geprint

Wat wordt er geprint?

Of kunnen we dat nu nog niet zeggen

… …

Volgens mij print hij de hele lijst

Dus hij zal een lijst geven met verschillende cijfers

Dus hij print cijfers

Nou, nee hij print de tekst, hij print eerst nummer 20 dat is groter dan heel koud dus dan print hij lekker thuis met een kop chocolademelk en dan vervolgens 5 print hij dat weer

En bij 4 is hij gelijk aan dus dan print hij lekker zonnen

En bij 15 print hij thuis

Ok, het is duidelijk

Ja

Top bedankt

[03:20]

{uitleg}

Top

Dan de tweede code

Deze is iets lastiger

[04:05]

Deze kennende

Als je weer hardop kan voorlezen

Definieer ik weet niet wat bbn a is

Ik denk dat het een willekeurige

Of hij print e maar e is 0

Als hij niet gelijk is aan c dus als a niet 0

Maar je weet volgens mij niet wat a is

Ok

Dus hij kan meteen e printen

Ok

Oh nee wacht

Ja hij kan meteen e printen

Ja ik weet niet wat bbn is

Of hij print gewoon e, dus dan is e 0

Of hij wordt wel geactiveerd, dan is a niet gelijk aan c

Deze kan sowieso niet, want hier staat een elif c is gelijk aan 0

Oh wacht hij kan wel, want c is 0

Oh wat raar

B is

De eerste voorwaarde is als a gelijk is aan 0

Die kan volgens mij niet want a is niet gelijk aan c

Aa moet 0 zijn anders dan gaat de loop niet in werking

Ok

Dus als a gelijk is aan 0

Dan doet hij e plus is 0 dat is volgens mij niet correct

Misschien heb ik daar geen ervaring mee, maar ik denk dat dat niet logisch is

Het is vast iets want het staat er twee keer in, dus misschien is het wel een functie die ik gewoon niet ken

Als b is kleiner dan a

Maar deze kan ook niet want als b kleiner dan a, dan is a gorter dan 2, en dan is a niet gelijk aan c

Dus dat is ook niet in de loop

Dus ik denk hij print gewoon e, want e is gewoon 0

Want de enige plek waarop e verandert, is e plus is 0

Maar voor zover ik weet is dat niks dus dan zou hij nog steeds 0 zijn

Ok bedankt

[06:40]

{uitleg}

Ok, dan de laatste code, veel succes

Ok

[08:10]

Name input, what is your name, ja na hij definieert de naam en achternaam als ik gok gewoon de input

Ok

Ok deze zijn allemaal lijsten kijk ik zo wel naar

Dit is een while loop, hij print uiteindelijk gewoon de hele list dat is dan mylist, de hele lijst

Last list

Mylist pop ik heb geen idee wat dat pop is dus dat is ook jammer

High, ja maximum dus hoogste getal

Place is de derde van mylist

Dan heb je while i kleiner of gelijk aan 2

Dus dan over het twee keer uit

Mylist append integer i keer i

De eerste keer is het gewoon 1 keer 1

Ja ik weet niet wat append is

Ik gok nee, ik heb eigenlijk geen idee

Mylist sort oh hij sorteert iets

Ok

Uiteindelijk print hij ik denk de gesorteerde lijst

Daarnaast print hij last pop

Ik denk ja

Stomme gok hij print de laatste, dus dat is dan 19, want hij is gesorteerd

Dan de hoogste dat zou dan ook 19 zijn

Ok

Dan de derde, 0, 1, 2 dat is nummer 2 volgens mij

Dat is wat ik denk

Ok bedankt

[10:20]

{uitleg}

Interview 13

Ik zal je drie stukjes code geven, je mag op het papier tekenen om aantekeningen te maken

De bedoeling is dat je aan mij uitlegt wat er wordt geprint, of er iets wordt geprint, meerdere dingen worden geprint of niks wordt geprint

Ok

En als je hardop kan voorlezen waar je mee bezig bent

Heb ik dan alles gezegd, volgens mij wel

Ok

Nou hier is de eerste code

[00:30]

Ok, het gaat sowieso over temperatuur

Warm of koud

Ook blijkbaar een leuke verkeerde spelling van Duitsland en Holland

Even kijken, het valt allemaal onder Nederland

For temperatuur in graden

Dit telt niet mee

If item is kleiner dan lekker warm, dus kleiner dan 4 wat 3 getallen zijn

Dan is het niet lekker weer, ok

Dus als het lager dan 4, dan code

Als het groter is dan heel koud, dan thuis

Als het er tussenin zit dan niet geweldig weer

Ja, dat eigenlijk

Dus wat zou eerst worden geprint

Dus als, wordt ooit het item gekozen

Wacht even hoor

Even kijken

Want is het dus random of is het gewoon

.. …

Dit wordt helemaal nooit meer gecalled, tenminste voor zover ik het zie

ok

Ik zal trouwens niet aangeven of je het goed doet, ik zal alleen maar zeggen ok

Is goed

Ehm ja voor wat ik zie, ik zie niet precies of het random is of niet

Dus ik ga er van uit dat het random is

Ok

Maar als er dus random gekozen wordt

Dan wordt erbij, onder 4 graden lekker warm gezegd wat heel interessant is

En boven de 14 graden lekker thuis en als het daar tussenin zit wordt er niet geweldig weer geprint

Ok top bedankt

[03:08]

{uitleg}

Dan het tweede stuk code, deze is iets lastiger. Maar veel succes

Ok

[03:45]

Ok … …

Als je weer hardop kan voorlezen waar je mee bezig bent

Oh ja,

Eerste wordt er een heel ding gedefined wat vervolgens gecalled wordt

Dus er wordt sowieso e geprint wat 0 is dat wordt sowieso gedaan, dat gebeurt altijd

Dus als a niet c is

.. ..

Even kijken

Als a niet c is

Er wordt ook geen a gegeven

Dus dat mist er sowieso dus deze hele loop kan niet worden gebruikt

Maar e wordt alsnog geprint, dus het enige wat wordt geprint is 0

Volgens mij

Ok

[05:00]

{uitleg}

Dan het laatste stukje code

[05:40]

Ok even kijken … …

Even kijken … …

Als i kleiner is dan 2 wat hij in het begin is, dan ik weet niet precies wat append betekent

Ehm even kijken., maar dan gaat hij dus dat getal keer zichzelf doen

Want dat wordt meestal gebruikt als keer

Dus als het kleiner is dan zichzelf dan gaat het keer en ook nog plus 1

Dus dan zou het hier eindigen op 2, want 1 keer 1 is 1

Daarna ga je dit hele rijtje sorteren, dat gaat op volgorde van 0 naar 1, van nul naar hoog

Ok

Whatever

Je hoeft het niet helemaal uit te werken

Het hoogste is 19 volgens mij

Ehm en dan nulde is altijd 0 is het laagste getal mylist verandert dan naar 0

Last is weet net precies wat pop is,

Ok

Even kijken, het hoogste is maximaal van mylist maar mylist is verandert naar 0, dus dan is het hoogste ook 0

Ok

… …

Index

… …

Dit wordt trouwens ook niet meer gecalled

Het gaat constant over mylist

Dan wordt het

Wat hier gebeurt vind ik een beetje apart bij mylist [0] want ik weet niet of het, het opnieuw definieert of dat het gewoon alleen het laat zien

Ok

Ehm, want vervolgens gebruikt hij nog steeds mylist, want bij place gebruikt hij mylist 3

Dat zou dan het 4de getal zijn dat zou dan 0, 1, 2, 3, dat zou dan getal 3 zijn

Dan print hij de hele reeks of getal

Dan print hij last, ik weet niet precies wat pop is, dus ik weet niet wat hij print daar

Ok

Bij hoog gaat hij dus of naar 19 of naar 0

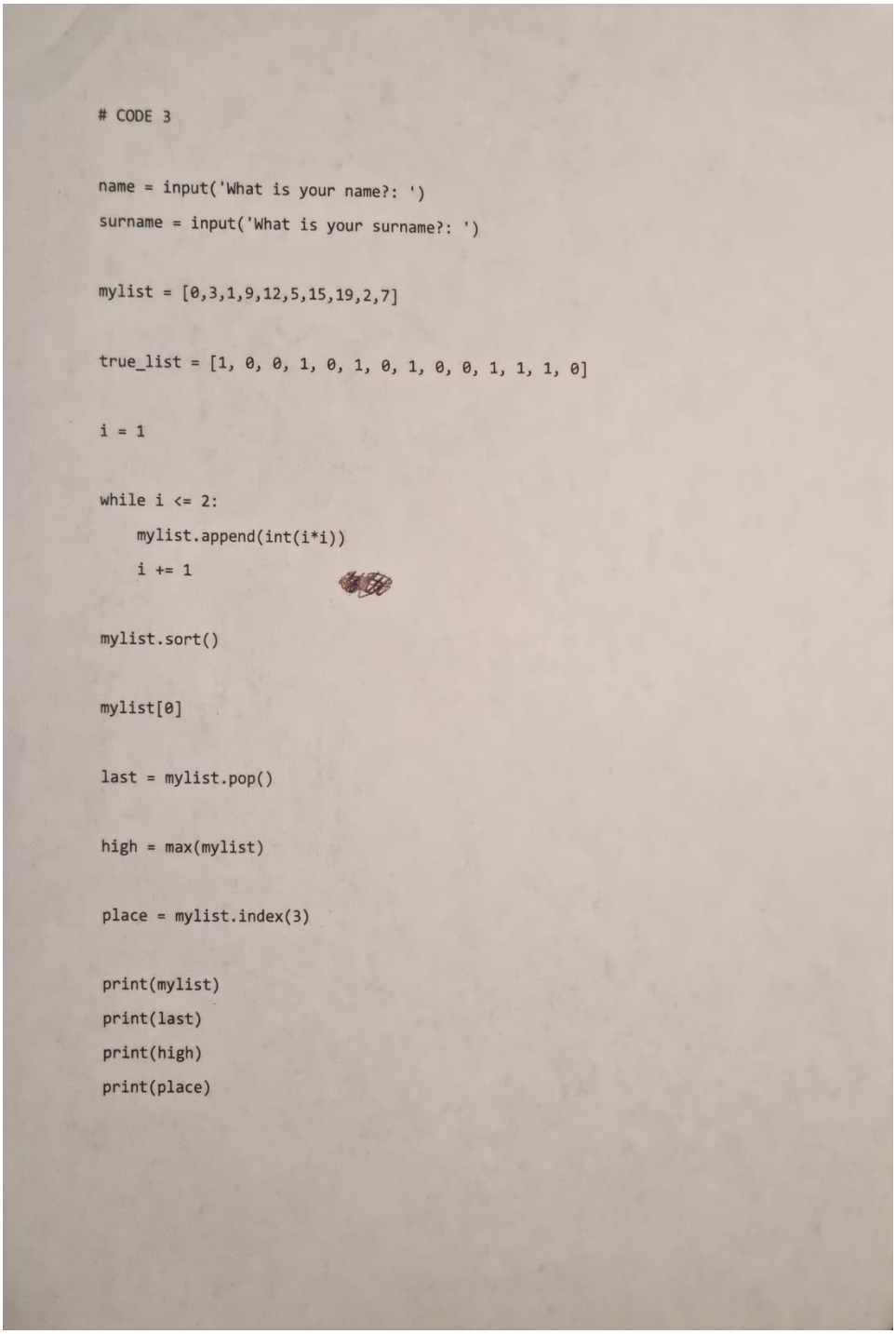
En bij place dus 0 of 3

Ok bedankt, top

[09:04]

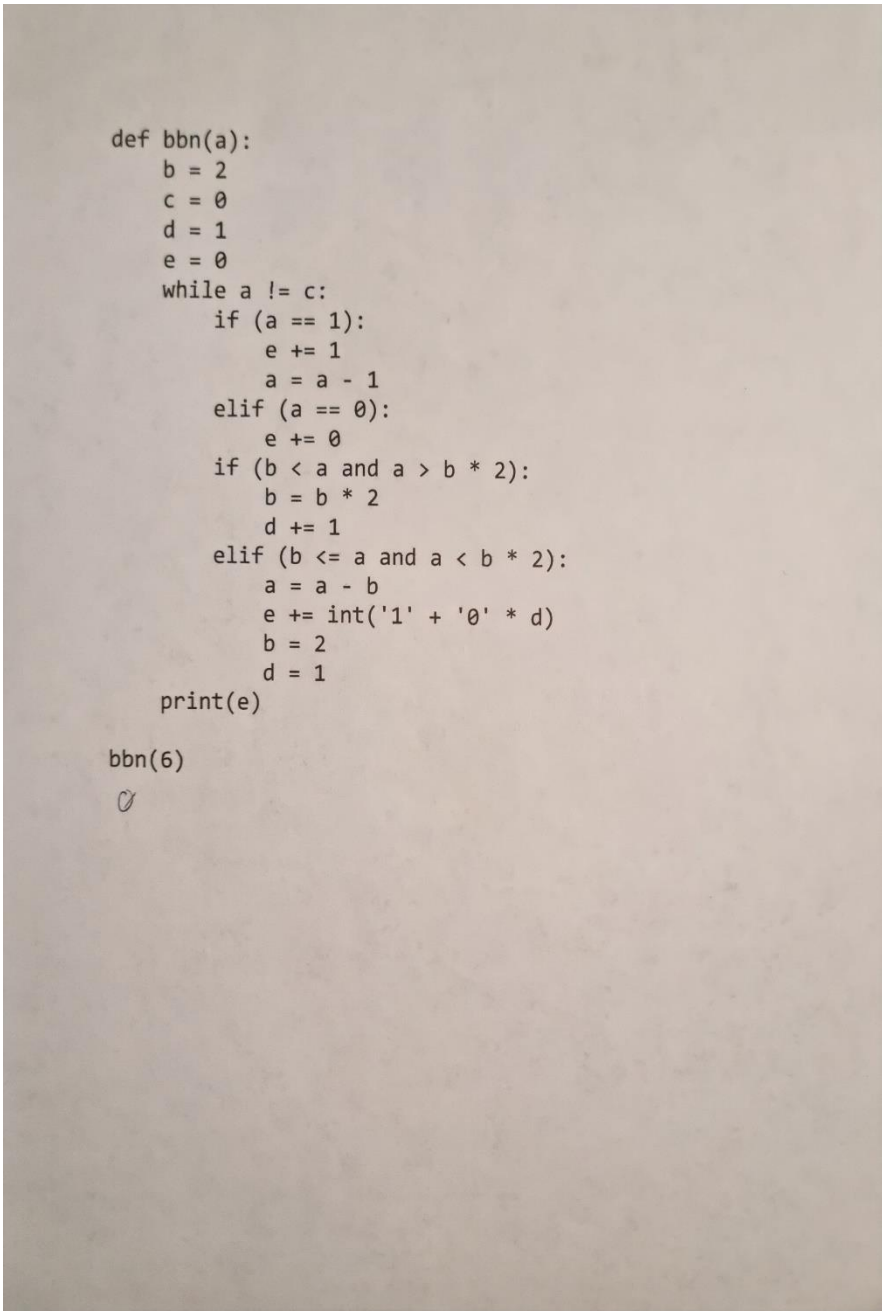
{uitleg}

10.3 Notes on code



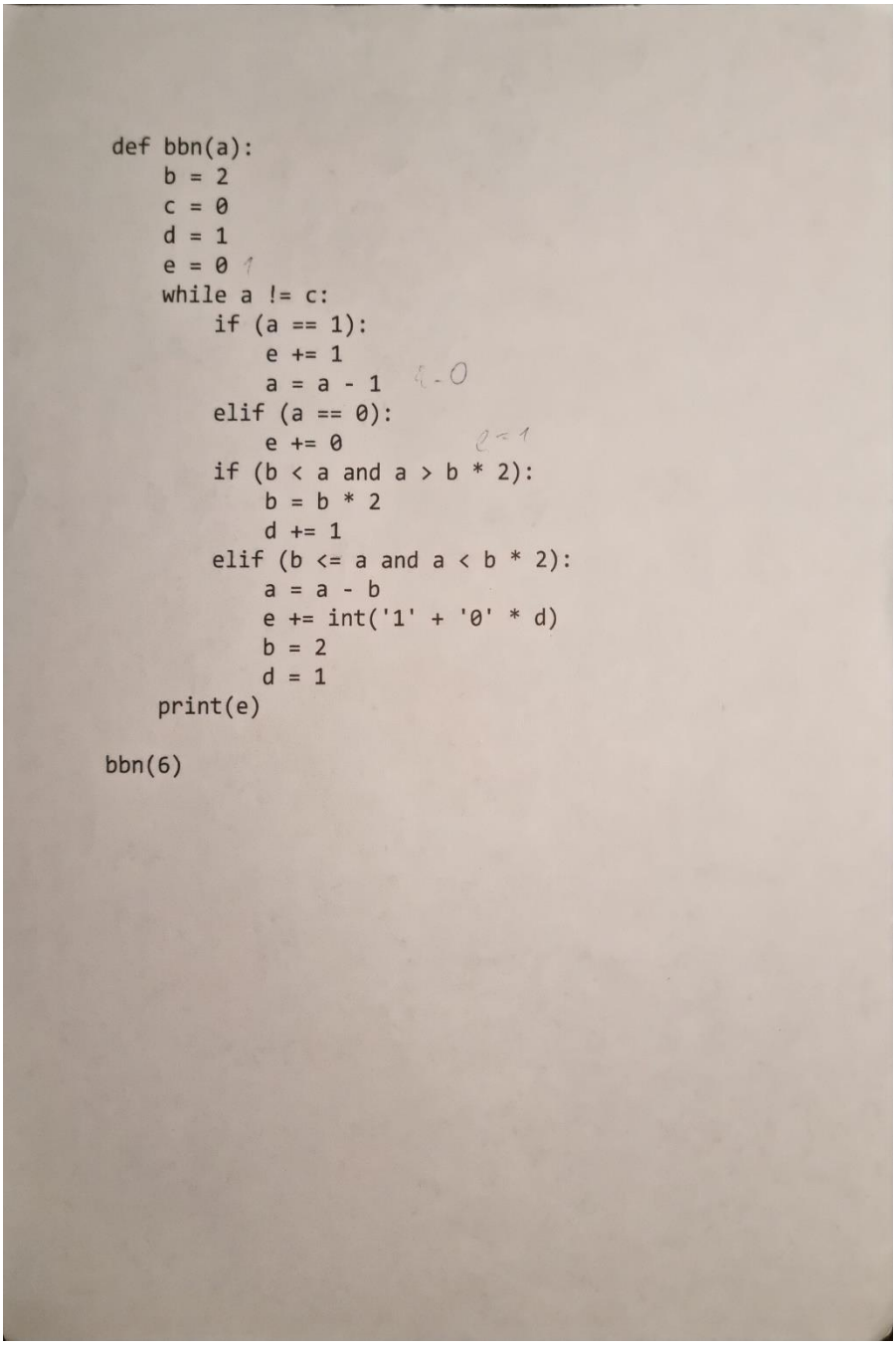
```
# CODE 3

name = input('What is your name?: ')
surname = input('What is your surname?: ')

mylist = [0,3,1,9,12,5,15,19,2,7]

true_list = [1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0]

i = 1

while i <= 2:
    mylist.append(int(i*i))
    i += 1

mylist.sort()

mylist[0]

last = mylist.pop()

high = max(mylist)

place = mylist.index(3)

print(mylist)
print(last)
print(high)
print(place)
```

Figure 6 - Notes on code



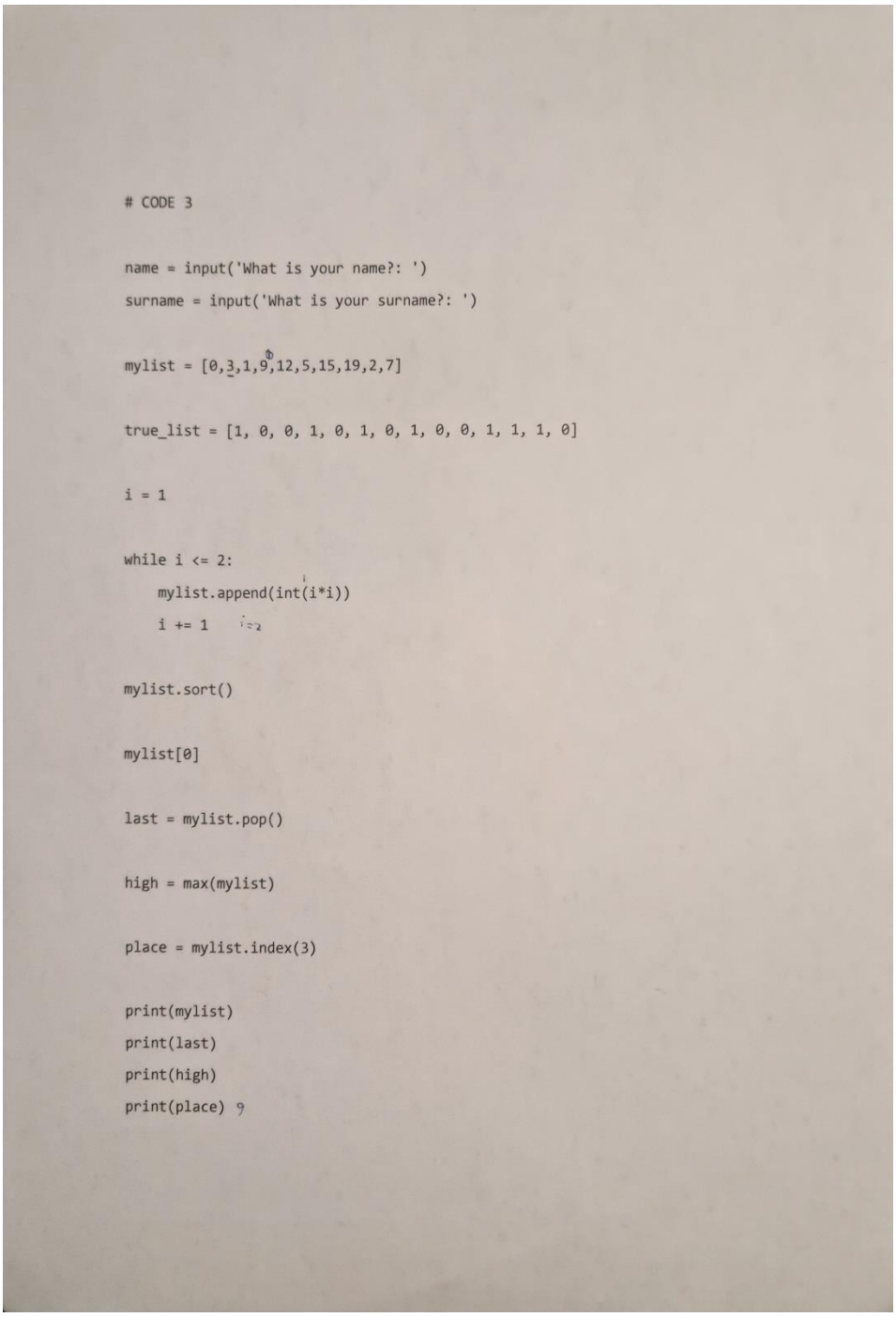
```
def bbn(a):
    b = 2
    c = 0
    d = 1
    e = 0
    while a != c:
        if (a == 1):
            e += 1
            a = a - 1
        elif (a == 0):
            e += 0
        if (b < a and a > b * 2):
            b = b * 2
            d += 1
        elif (b <= a and a < b * 2):
            a = a - b
            e += int('1' + '0' * d)
            b = 2
            d = 1
    print(e)

bbn(6)
```

Figure 7 - Notes on code



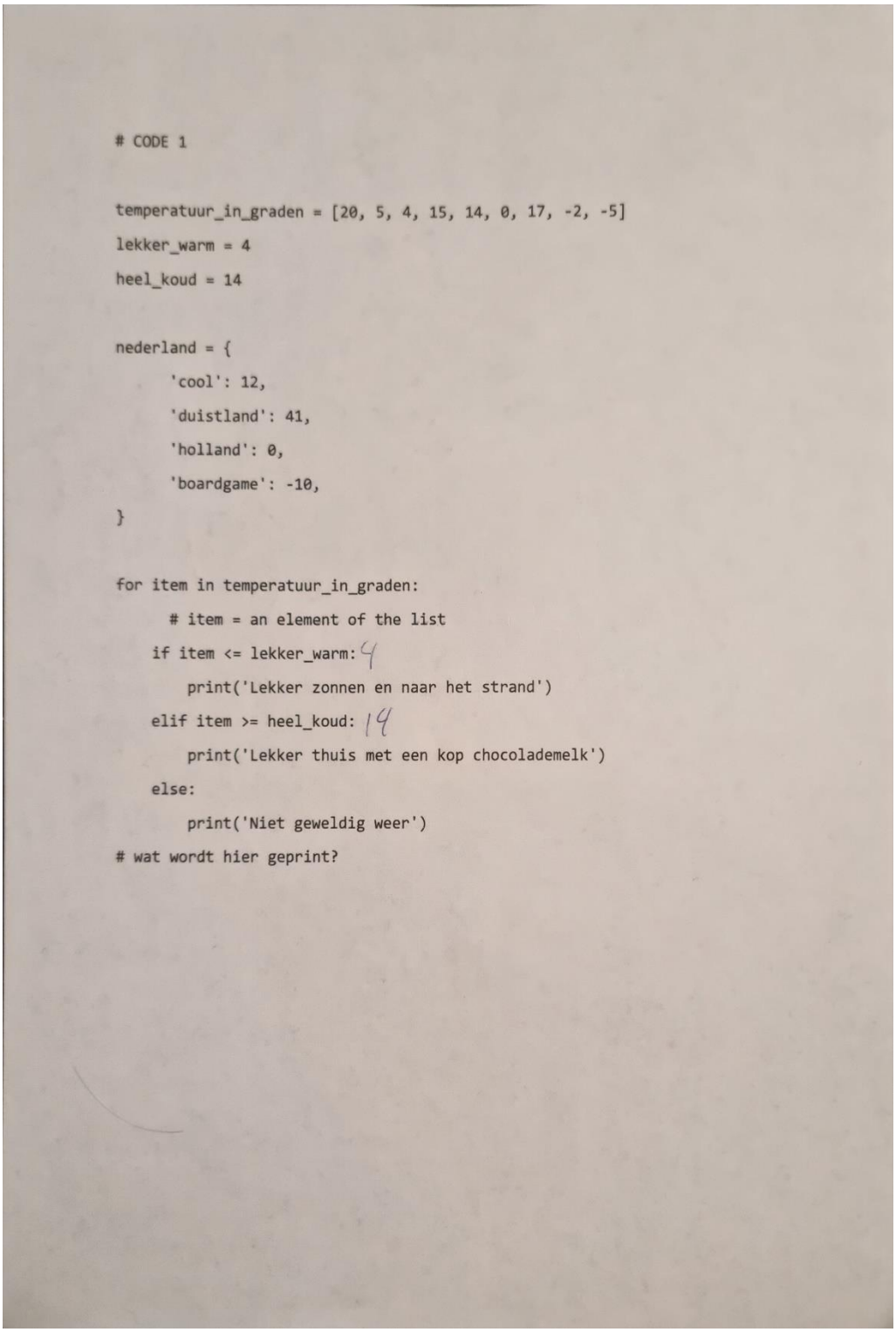
```
def bbn(a):
    b = 2
    c = 0
    d = 1
    e = 0
    while a != c:
        if (a == 1):
            e += 1
            a = a - 1
        elif (a == 0):
            e += 0
        if (b < a and a > b * 2):
            b = b * 2
            d += 1
        elif (b <= a and a < b * 2):
            a = a - b
            e += int('1' + '0' * d)
            b = 2
            d = 1
    print(e)

bbn(6)
```

Figure 8 - Notes on code



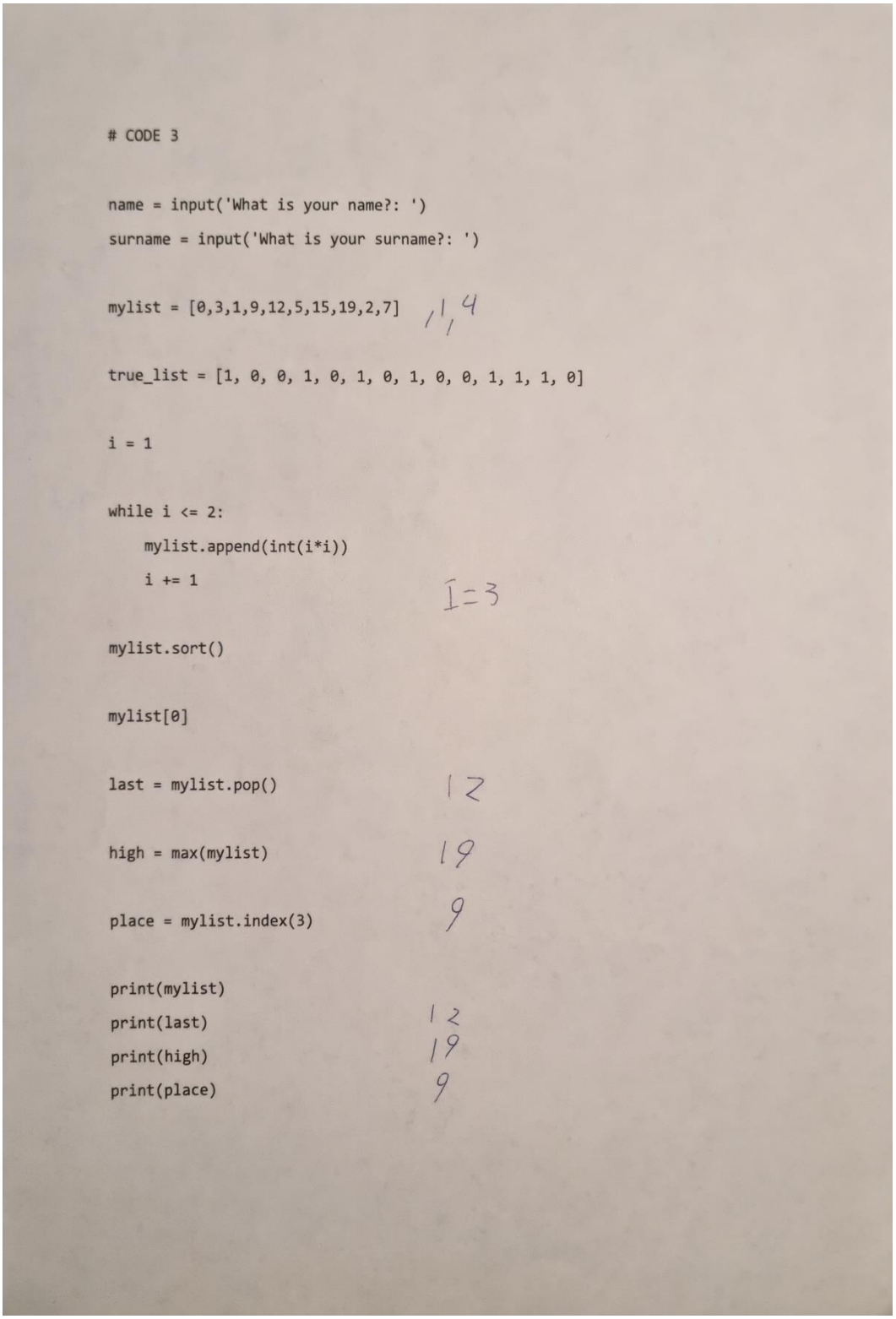
```
# CODE 3

name = input('What is your name?: ')
surname = input('What is your surname?: ')

mylist = [0,3,1,9,12,5,15,19,2,7]

true_list = [1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0]

i = 1

while i <= 2:
    mylist.append(int(i*i))
    i += 1

mylist.sort()

mylist[0]

last = mylist.pop()

high = max(mylist)

place = mylist.index(3)

print(mylist)
print(last)
print(high)
print(place)
```

Figure 9 - Notes on code



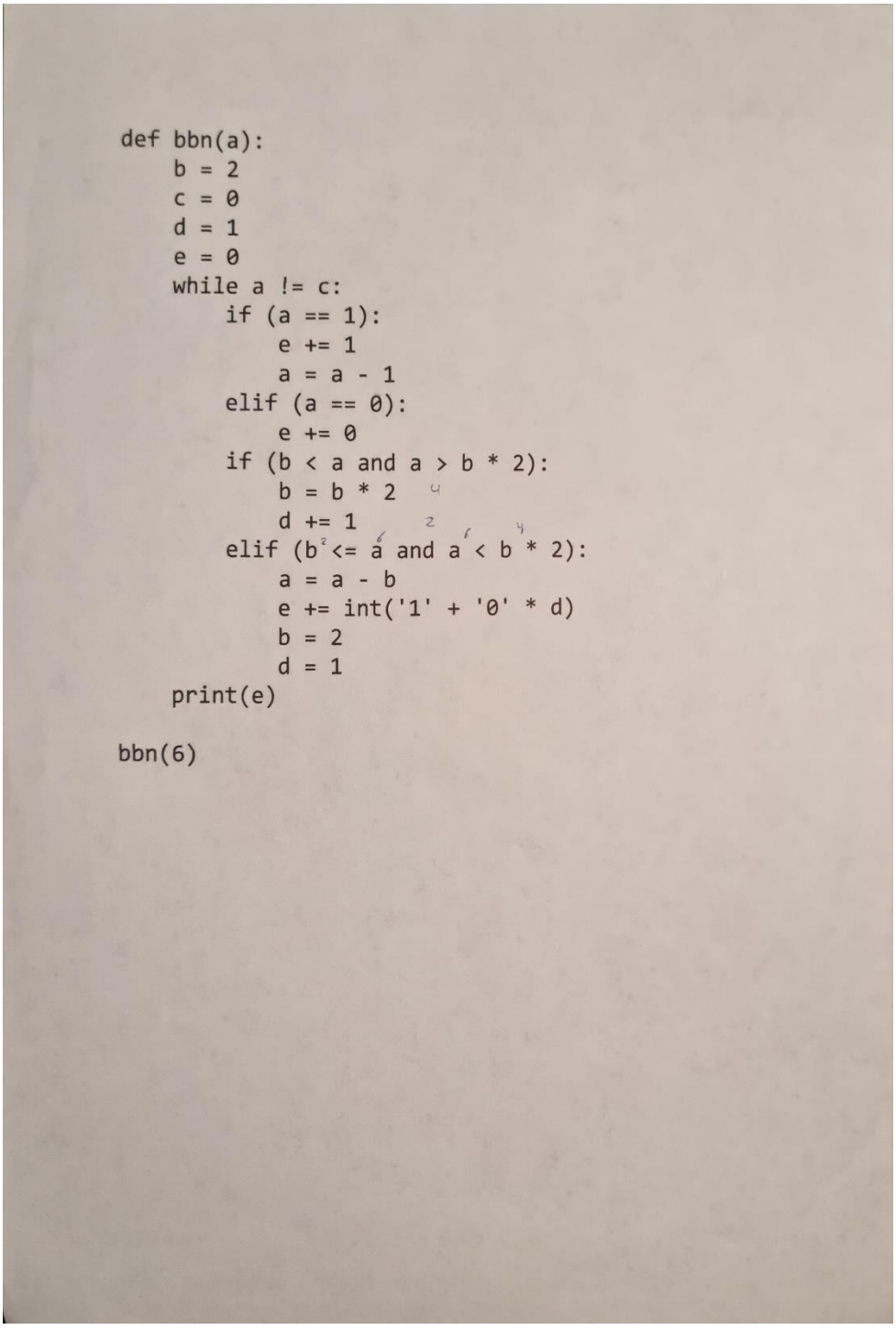
```
# CODE 1

temperatuur_in_graden = [20, 5, 4, 15, 14, 0, 17, -2, -5]
lekker_warm = 4
heel_koud = 14

nederland = {
      'cool': 12,
      'duistland': 41,
      'holland': 0,
      'boardgame': -10,
}

for item in temperatuur_in_graden:
      # item = an element of the list
    if item <= lekker_warm: 4
        print('Lekker zonnen en naar het strand')
    elif item >= heel_koud: 14
        print('Lekker thuis met een kop chocolademelk')
    else:
        print('Niet geweldig weer')
# wat wordt hier geprint?
```

Figure 10 - Notes on code



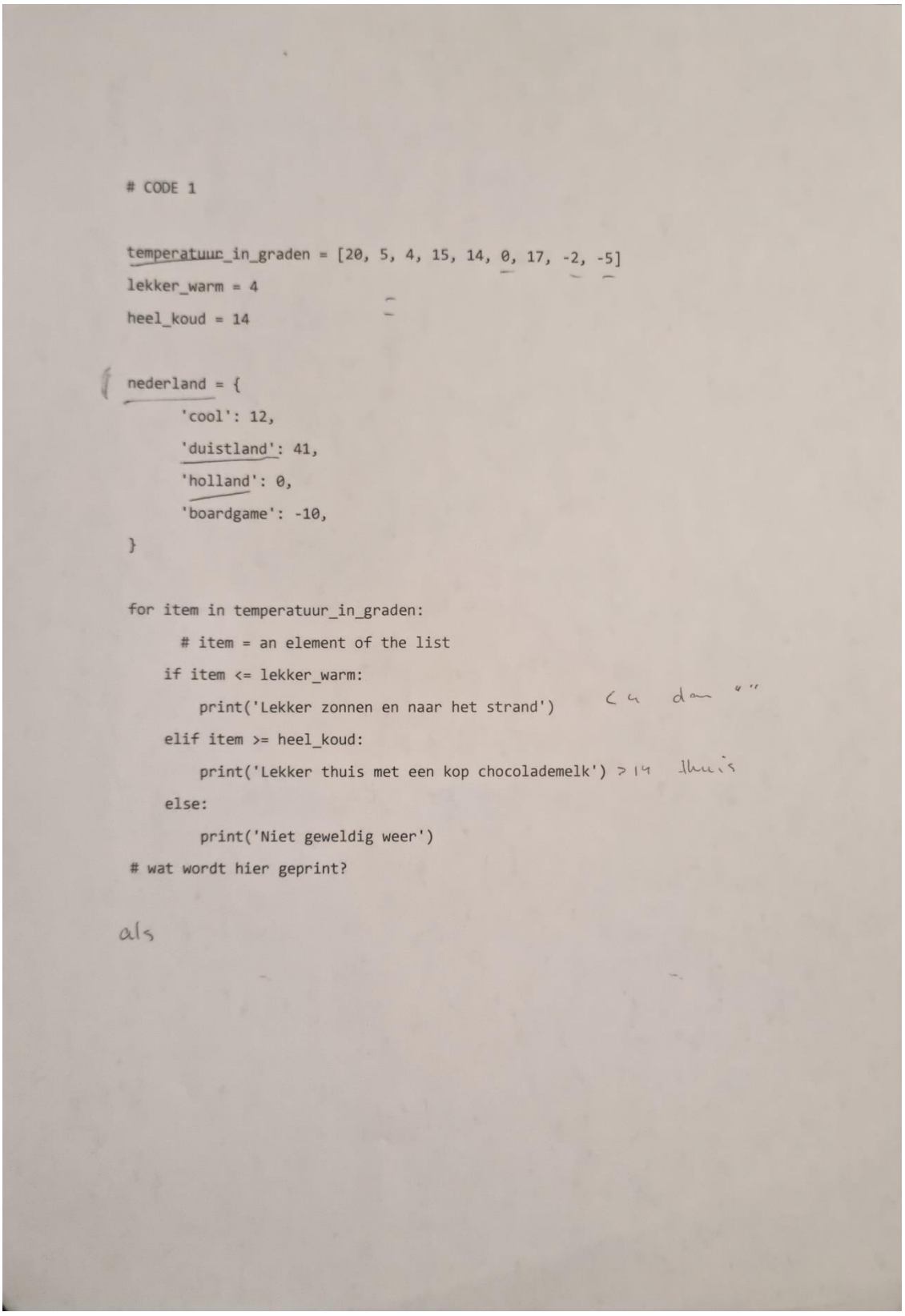
```
# CODE 3

name = input('What is your name?: ')
surname = input('What is your surname?: ')

mylist = [0,3,1,9,12,5,15,19,2,7]

true_list = [1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0]

i = 1

while i <= 2:
    mylist.append(int(i*i))
    i += 1

mylist.sort()

mylist[0]

last = mylist.pop()

high = max(mylist)

place = mylist.index(3)

print(mylist)
print(last)
print(high)
print(place)
```

Figure 11 - Notes on code



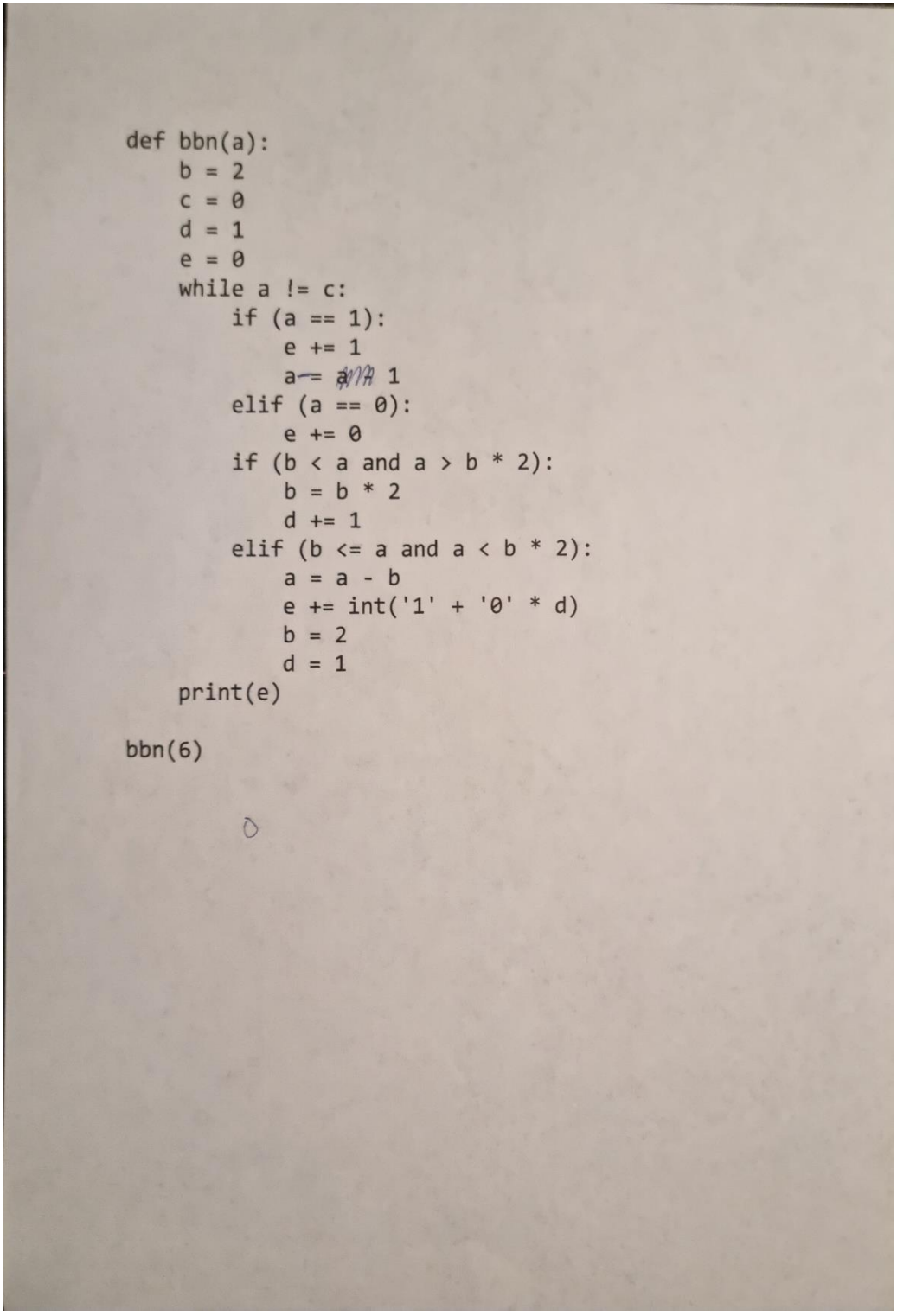
```
def bbn(a):
    b = 2
    c = 0
    d = 1
    e = 0
    while a != c:
        if (a == 1):
            e += 1
            a = a - 1
        elif (a == 0):
            e += 0
        if (b < a and a > b * 2):
            b = b * 2
            d += 1
        elif (b <= a and a < b * 2):
            a = a - b
            e += int('1' + '0' * d)
            b = 2
            d = 1
    print(e)

bbn(6)
```

Figure 12 - Notes on code



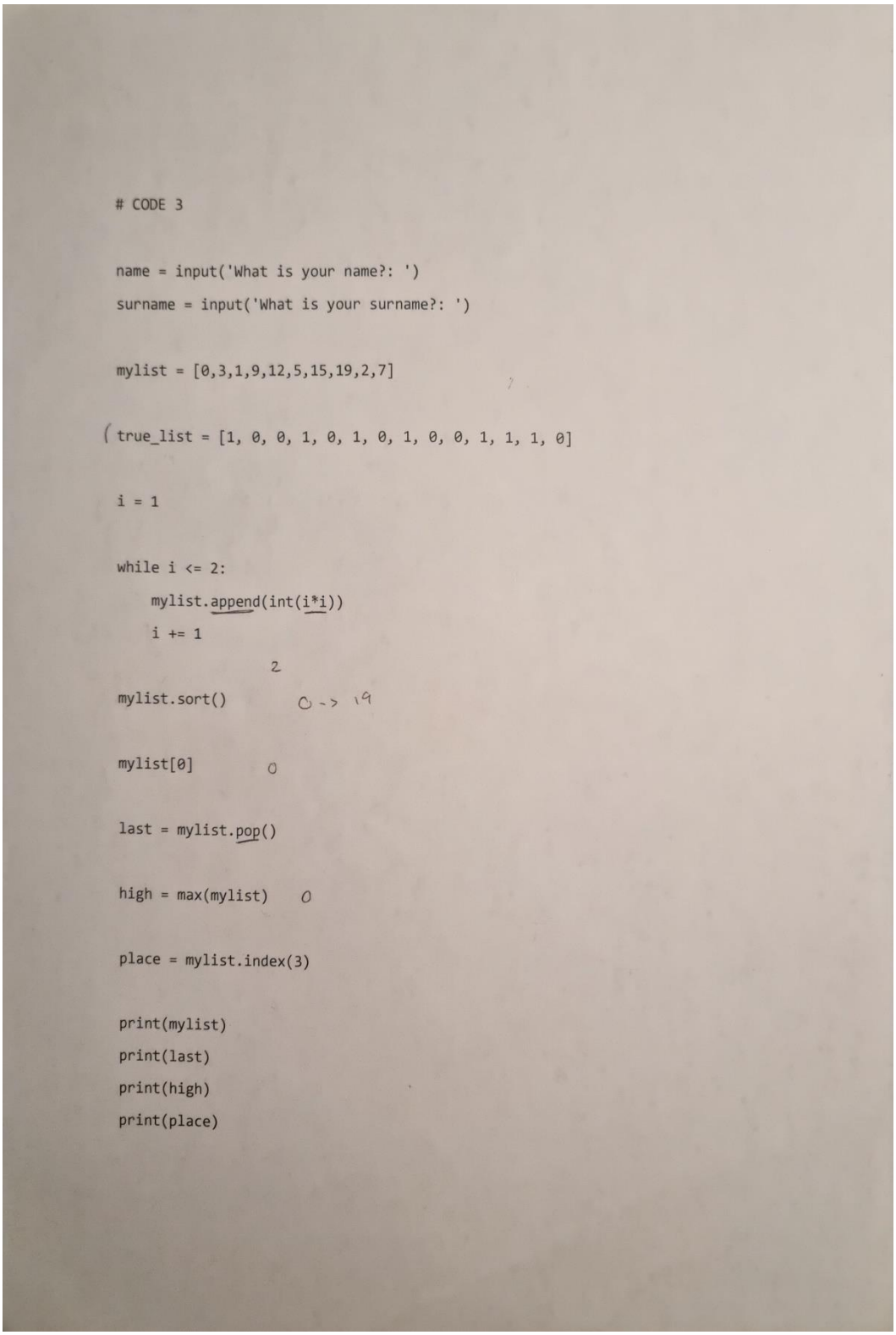
```
# CODE 1

temperatuur_in_graden = [20, 5, 4, 15, 14, 0, 17, -2, -5]
lekker_warm = 4
heel_koud = 14

nederland = {
      'cool': 12,
      'duistland': 41,
      'holland': 0,
      'boardgame': -10,
}

for item in temperatuur_in_graden:
      # item = an element of the list
    if item <= lekker_warm:
        print('Lekker zonnen en naar het strand')
    elif item >= heel_koud:
        print('Lekker thuis met een kop chocolademelk')
    else:
        print('Niet geweldig weer')
# wat wordt hier geprint?

als
```

Figure 13 - Notes on code



```
def bbn(a):
    b = 2
    c = 0
    d = 1
    e = 0
    while a != c:
        if (a == 1):
            e += 1
            a -= 1
        elif (a == 0):
            e += 0
        if (b < a and a > b * 2):
            b = b * 2
            d += 1
        elif (b <= a and a < b * 2):
            a = a - b
            e += int('1' + '0' * d)
            b = 2
            d = 1
    print(e)

bbn(6)
```

Figure 14 - Notes on code



```
# CODE 3

name = input('What is your name?: ')
surname = input('What is your surname?: ')

mylist = [0,3,1,9,12,5,15,19,2,7]

true_list = [1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0]

i = 1

while i <= 2:
    mylist.append(int(i*i))
    i += 1

mylist.sort()

mylist[0]

last = mylist.pop()

high = max(mylist)

place = mylist.index(3)

print(mylist)
print(last)
print(high)
print(place)
```

Figure 15 - Notes on code