My memories of the Programming-assignment Winter 2022

General remark

I did not have enough time! But: they said right from the beginning that the exam was designed in the way that not enough time was given. One could skip an enitre exercise and still make some minor errors, and still be able to achieve the best grade. So the exam had not many tricky and complicated exercises, but instead, it was like: "how fast are you and how many mistakes do you make when programming under time pressure?". Also, it is worth mentionning that they did accept solutions which could not be compiled, but: In this case, the number of points which could be obtained for the corresponding exercise was capped at 75% of the original maximal points. So a solution of a 15-points-exercise could maximally earn 10 points, if the compilation failed. By the way: all exercises gave around 10 to 15 points.

I would give three advices: Be sovereign with implementing different classes. 3 of 9 exercises requirered the participants to implement several classes at once. Though these classes tended to be rather simple. And the second advice: have a rough idea about the most important String-methods. Finnally, I would say that for most exercises, the LIVE assignments were a suited preparation.

Exercise 1

Several code fragements like the following were given:

```
int a = 5;
int b = 3;
value = a + b;
```

One should say, of which data type `value` had to be in order that the code works without error. In case that an error was unavoidable, one should instead point out that the code will fail and explain why. An example of this is the following:

```
int a = 5;
int b = "3";
value = a + b;
```

In this case, since the code will fail in the second line, pointing that out would be correct. The "errors" in the exam were a bit more tricky, however. E.g. calling a wrong method.

Exercise 2

A rather simple function was given, that contained a for-loop. The task was to write a new function, that performs the exact same task, but does NOT contain a for-loop.

This exercise could be solved by simply replacing the for-loop with a while-loop.

Exercise 3

The topic here were strings. A function which reads in two strings and counts, how often the second string occured in the first, were to be implemented. Upper and lower cases were to be ignored.

E.g. if the first input string were "Jojo" and the second was "j", the function should output 2, since j occures two times in "Jojo", if we make no difference between upper and lower cases (hence "j"="J").

Exercise 4

The topic here was recursion. A function were to be implemented which read in an int-array and an index. It should work as follows (in the exam, it was also described that way):

$$g(a,i) = \begin{cases} ''b'' + g(a, a[i]); \text{ if a[i] is valid and is even} \\ ''a'' + g(a, a[i]); \text{ if a[i] is valid and is odd} \\ '.'; \text{ if a[i] is not valid (beacuse a[a[i]] cannot be computed)} \end{cases}$$

Example: if the array was {1, 3, 5, 2, 1} and the start index was 1, the function should via recursion yield the following result: "ab."

Other example: the same array with the start index 0 should yield: "bba."

Exercise 5

A function was to be written which reads in an 2-dimensional array of doubles (`double[][]`). This array represented a matrix with an unknown number of rows and 12 columns (each column was a month). The function should output a vector, containing the average of every row. (The "background-story" of the matrix was: in each row, for a fixed year, the average temperatures of the corresponding months were registered).

Exercise 6

The topic was "collections". Defining some classes that interact with each other was needed. I don't remember more detail.

Exercise 7

I think it was about exception handling. The task was to write a class `recpet`, that fulfilled some properties (not complicated). The main part was to define a method, that read in a string for a recept, like the following: "Pizza Hawaii 10.50", were the number in the end was the price. So the string should contain the Information "name" + "price" in exactly this order. The method should create an object of class recipe from this information.

However, the programm shall not crush even if the user inputted nonsence, like "Pizza 10.50 Hawaii".

Exercise 8

It was about inheritrance. An abstract "superclass" were to be defined, followed by some classes, that inheriated methods from it. E.g., the superclass could be "vehicle" and the lower classes could be "car", "bike", "bus".

In the task, some methods were mentionned that were to be implemented.

Exercise 9

Something concerning Queues. I didn't attempt solving this exercise, but it reminded me at what we have had in the live assignments. A class "Queue" was GIVEN and a class "Elem". But the class "Queue" was sligthly differently implemented than what we've seen in the assignments.