Prof. Dr. Visvanathan Ramesh
Alperen Kantarcı
Gamze Akyol

Goethe-Universität Frankfurt am Main
Institut für Informatik

Programmierparadigmen und Compilerbau

Sommersemester 2023

Exercise Nr. 1

Deadline Wednesday 10.05.2023, 10:00 in Moodle!

Question 0 (0 Points)

Read the following tips!

- Group work is **not allowed!**. You can talk and discuss with your friends about the exercises, but you should solve the exercises **individually**. Plagiarism is not tolerated.
- You should always test your functions and you should put either a test function or a comment that shows a valid function call inside your code.
- You can use ChatGPT or other web sources but you cannot ask the whole question. You should be able to explain every step of your code.
- Upload all the files you want to submit into **a single .zip file on Moodle**. Please ensure that you **only submit .pdf and .hs files.**
- Please write your **matrikelnummer name**, **group number** and to each file that you are submitting.
- Your submission file name should be in the following order:

  Name_Matrikelnummer_Group_GroupNumber_Exercise_ExerciseNumber.extension.

  For example **AlperenKantarci_1111111_Group_2_Exercise_1.zip**

Question 1 (6 Points)

Universal Product Code (UPC) is a barcode type widely used in stores. In this question you will validate UPC-E barcodes. UPC-E is a smaller barcode for using UPC on smaller packages. It consists of the number system, 6 digits and a check digit. You can see an example below.

Abbildung 1: Example UPC-E barcode. Image source: `https://docs.devexpress.com/OfficeFileAPI/15148/barcode-generation-api/bar-code-types/upc-e0`

Check digit is a digit calculated from the 6 digits code and we will not consider number system. In order to calculate check digit, following algorithm should be executed:

1. Starting from the right, sum the odd digits.

2. Multiply your result from step one with 3
3. Starting from the right, sum the even digits.
4. Add results of step two and step three
5. Divide the result of step four by 10. The check digit is the number which adds the remainder to 10.

For example, for code 425461 the check digit would be 4. Sum of odd digits would be (1+4+2) = 7. Sum of even digits would be $(6+5+4) = 15$. After multiplying odd sum with 3 and adding even sum, result would be $7 * 3 + 15 = 36$. Dividing 36 to 10 would give us remainder 6. So our check digit is $10 - 6 = 4$.

You will write a function calculateCheckDigit which takes 6 digit UPC code and returns the check digit.

Question 2 **(3+3 = 6 Points)**

An anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once. For example, the anagram of the "cat" word is as follows: ["cat", "act", "tac", "cta", "tca", "atc"]

1. Write a function that takes in a string and returns a **lazy** list of all possible anagrams of the string, generated using a lazy permutation algorithm. (**Hint:** Think about the difference between lazy evaluation and strict evaluation and try to answer from this perspective.)
   Example: Input: "cat"
   Output: ["cat", "act", "tac", "cta", "tca", "atc"]
2. Write a function longestAnagram :: [String] -> String that takes a list of strings and returns the longest string that is an anagram of another string in the list.
   Example: Input: ["listen", "silent", "tac", "cat", "rats", "arts", "stab"]
   Output: "listen"

Question 3 **(1+1+2+2+2 = 8 Points)**

In this question, you will implement different functions to find absolute difference between the sum of the squares of the first N natural numbers and the square of the sum. For each step you can use the functions you wrote in earlier steps.

1. Write a function absdiff :: Int->Int->Int which returns the absolute difference between two integers
2. Write a function square :: Int->Int which returns n^2 for the given integer n.
3. Write a function sumOfSq :: Int->Int which takes integer N and returns the sum of squares for first N natural numbers. For example for N=4 it should return $1^2 + 2^2 + 3^2 + 4^2 = 30$
4. Write a function squareOfsum :: Int->Int which takes integer N and returns the squares of the sum for first N natural numbers. For example for N=4 it should return $(1+2+3+4)^2 = 100$
5. Write a function squareDiff: Int->Int which takes integer N and returns the absolute difference between the sum of the squares of the first N natural numbers and the square of the sum.