C++ Programming

Exercise Sheet 01
Software Engineering, EIM-I
Philipp Schubert
October 20, 2017

Solutions to this sheet are due on 27.10.2017 til 14:00. Please hand in a digital version of your answers via e-mail. The e-mails subject has to contain ccpp. Do zip-compress your solutions. For questions please send mail or speak to me during the exercises.

Note: If you copy text elements / code elements from other sources, clearly mark those elements and state the source. Copying solutions from other students is prohibited. All of your files that belong to your solution have to be contained in a single .zip file that is named according to the following naming scheme: <name>_<surname>_solution<XX>.zip. Replace <name> and <surname> with your actual name and replace <XX> with the sheet number the solutions belong to. You can look up your results using this link [https://docs.google.com/spreadsheets/d/1V8rKtimsQS6thKGKTh6CChlv-LwulBIA3rvkA1ZSH2M/ edit?usp=sharing](https://docs.google.com/spreadsheets/d/1V8rKtimsQS6thKGKTh6CChlv-LwulBIA3rvkA1ZSH2M/edit?usp=sharing).

During this exercise sheet you will make yourself familiar with the different kinds of control flow that exists in C++. Furthermore you will start learning how to express mathematical and real-world problems in the C++ language. This is the first step of developing a computational thinking. You can achieve 16 points in total.

Exercise 1.

a) Write a program that reads an integer from the command line and checks if the value is greater than or equal to 0 and smaller than or equal to 100! The program should print the result of the check to the command line. (2 P.)

b) Write another program that reads an integer from the command line and checks if the integer
   1. is greater than 0
   2. holds check 1. and is additional dividable by 4
   3. holds check 2. and is additional dividable by 3

Print which of the checks hold for the integer. (Hint: use nested if statements) (2 P.)

Exercise 2.

Obviously you now would like to analyze sequenced DNA. DNA is (usually) made up from four different kinds of bases: guanin 'G'/'g', adenin 'A'/'a', cytosin 'C'/'c' and thymin 'T'/'t'. The following std::string with the name dna stores the DNA sequence you would like to analyze. Until now
we only had used strings as literals and never defined a variable of type `string`. We will learn about the non-build-in string datatype in the next lecture. For this exercise it is sufficient for you to know that you have to include the header-file `string` and how to iterate a string which is shown in the code below. Use the code snippet in order to solve the next few tasks. You can download the code snippet here [https://www.hni.uni-paderborn.de/fileadmin/Fachgruppen/Softwaretechnik/Lehre/CPP_Programming/WS2017_2018/code_01.zip](https://www.hni.uni-paderborn.de/fileadmin/Fachgruppen/Softwaretechnik/Lehre/CPP_Programming/WS2017_2018/code_01.zip).

```cpp
#include <iostream>
#include <string>
using namespace std;

int main() {
    const string dna = "AGTccaaGTCAGACATGAAAtataAATCG";
    // this range−for loop iterates the string 'dna'
    for (char base : dna) {
        // you can use the variable 'base' inside this loop
    }
    return 0;
}
```

a) Iterate `dna` and use a switch statement in order to count the occurrences of each of the four different DNA bases! Use one counter variable for each base. Print the number of occurrences for each base on the command line! (1 P.)

b) Extend your program so that it is able to read a string from the command line! You can use the `string` data type in combination with `cin` just like you did for the build-in data types. (1 P.)

c) As a final extension to your DNA-processing-program, add some functionality that tells the user how many DNA bases are encoded as lower case letters and how many are encoded as upper case letters! (1 P.)

d) In which cases should you prefer a `switch` statement over an `if` statement and why? (1 P.)

Exercise 3.

Consider the following program that performs a simple numeric integration of \[ \int_{0}^{1} 4 / (1 + x^2) \, dx \] and prints the result on the command line. The source code of this program can be found here [https://www.hni.uni-paderborn.de/fileadmin/Fachgruppen/Softwaretechnik/Lehre/CPP_Programming/WS2017_2018/code_01.zip](https://www.hni.uni-paderborn.de/fileadmin/Fachgruppen/Softwaretechnik/Lehre/CPP_Programming/WS2017_2018/code_01.zip). Compile and run the program.

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main() {
    const long double from = 0.0;
    const long double to = 1.0;
    long double integral_val = 0.0;
    long double x = from;
    const size_t N = 1000000;
    const long double step_width = abs(from − to) / static_cast<long double>(N);
    for (size_t n = 0; n < N; ++n) {
        integral_val += 4 / (1 + x*x);
        x += step_width;
    }
    return 0;
}
```
integral_val = integral_val / N;
cout << integral_val << "\n";
return 0;
}

a) Modify and use the above numerical integrator program such that it calculates \( \int_0^1 3x^2 \, dx \). (1 P.)
b) Modify and use the above numerical integrator program such that it calculates \( \int_0^1 2\sqrt{x} \, dx \). (1 P.)
c) Write a small program that computes \( \sum_{k=1}^{100} k \). (1 P.)
d) Write another small program that computes \( \sum_{i=1}^{10} (\sum_{j=1}^{10} i) \). (1 P.)

**Exercise 4.**
Write a program that prints the pattern to the command line which are shown in

a) [figure 1] (1 P.)
b) [figure 2] (1 P.)
c) [figure 3] (2 P.)

Hint: Combine nested loops and if statements! (All figures are 10 × 10 characters.)

```plaintext

Figure 1: Pattern A.  Figure 2: Pattern B.  Figure 3: Pattern C.
```