Chair for Clinical Bioinformatics
Pascal Hirsch
Annika Engel
Saurabh Pandey
Ilia Olkhovskii

Programming Course
1st Assignment

Hand in until 22.11.2023 at 11:59 pm

General remarks

Please make sure that you follow all rules stated in the general remarks PDF in the CMS. For this assignment you are allowed to assume that your code will only be tested with valid input. The provided **example** inputs are only **examples**. Your code should work with **any** other input as well. When reading from the standard input you can either first read the complete input until the stream ends or, if possible, handle the input as soon as you have the required data. The number of barbells 🏋 for each task describes the expected difficulty. The maximum of number of barbells per task will be 4.

Task 1 (20 P.): 🏋

1. List 8 built-in types provided by C++. (2 P.)
2. What is the difference between an expression and a statement? Provide an example for each. (1 P.)
3. What is type safety and why is it important? (1 P.)
4. When would a programmer prefer a switch-statement over an if-statement? (0.5 P.)
5. What is a common problem with switch statements? Provide an example. (1 P.)
6. Name one operation available for strings but not for integers and one operation available for integers but not for strings. (1 P.)
7. Explain what are references and pointers. Provide examples for each. (2 P.)
8. Explain the relationship between pointers and arrays (and C-strings). Provide examples. (2 P.)
9. Explain the pass by value and pass by reference concepts. Provide examples. (1 P.)

10. Explain the terms declaration and definition. (1 P.)
11. What is a namespace and what is its typical use? (1 P.)
12. Why should you avoid using directives in a header? (0.5 P.)
13. List four kinds of errors and provide an example for each. (2 P.)
14. What is the free store and for what reasons might one use it? (1 P.)
15. How do you allocate memory and about what do you need to be careful? Provide an example. (1 P.)
16. How do you safely allocate/manage memory? (1 P.)
17. What is RAII? (1 P.)

Task 2 (10 (5+5) P.):⤡+⤡

We provide two code snippets for this task in task2a.cpp and task2b.cpp. First compile and run the code. Then describe and explain the observed behaviour. Finally fix the code so that it behaves as expected.

Task 3 (5 P.):⤡

Given are multiple sets of space separated numbers on the standard input, each on their own line. Your task is to output to the standard output the median, mean, sum, standard deviation, first quartile and third quartile, each separated by one space and for every new set by a newline. Define functions for each measure. The quartiles should be computed as specified in the first method on Wikipedia (https://en.wikipedia.org/wiki/Quartile). If a number has more than 2 decimal places then round it to 2 decimal places, but leave other numbers unchanged.

Sample input:

```
3 5.2 10 -2
4 0 50 66 1
```

Sample output:

```
4.1 4.05 16.2 4.98 0.5 7.6
4 24.2 121 31.4 0.5 58
```

Task 4 (5 P.):⤡⤡

This task is an extension to *Task 3* and consider the same input. Define another function that scales the values for each row. The scaling should be implemented in such a way that every number should be divided by the sum of the row. Your output to the standard output should be space separated and on a new line for each row.

If a number has more than 2 decimal places then round it to 2 decimal places, but leave other numbers unchanged.

Sample input:

```
3 5.2 10 -2
4 0 50 66 1
```

Sample output:

```
0.19 0.32 0.62 -0.12
0.03 0 0.41 0.55 0.01
```

Task 5 (15 P.):

Given are sets of space separated numbers on the standard input, each on their own line, representing a square matrix. Your task is to output to the standard output the median, mean, sum, first quartile and third quartile (calculate according to the first method from (https://en.wikipedia.org/wiki/Quartile)) for every row and column of the matrix, as well as for its diagonal elements and for the elements in the lower triangle (without the diagonal elements). Your output should be space separated and on a new line for each row, column, the diagonal and lower triangle. To present the matrix use `vector<vector<string>>`. If a number has more than 2 decimal places then round it to 2 decimal places, but leave other numbers unchanged.

Sample input:

```
1 2 0
3 2.1 2
-5 4 8
```

Sample output:

```
  1 1 3 0 2
  2.1 2.37 7.1 2 3
  4 2.33 7 -5 8
  1 -0.33 -1 -5 3
  2.1 2.7 8.1 2 4
  2 3.33 10 0 8
  2.1 3.7 11.1 1 8
  3 0.67 2 -5 4
```

Task 6 (5 P.):

This task is an extension to *Task 5* and consider the same input. Define another function that standardises the values for each column. The standardisation should be implemented using the z-score. Your output to the standard output should be space separated and on a new line for each row. If a number has more than 2 decimal places then round it to 2 decimal places, but leave other numbers unchanged.

Sample input:

```
1 2 0
3 2.1 2
-5 4 8
```

Sample output:

```
0.32 -0.62 -0.8
0.8 -0.53 -0.32
-1.12 1.15 1.12
```

Task 7 (10 P.):

Write a program that takes a **FASTA** file as first argument and checks if the sequences are DNA or RNA and convert all DNA sequences into RNA and treat them as such in the following. If a sequence is neither DNA or RNA ignore it for all outputs and calculations. Further output the number of lines, the number of sequences, the GC content and the percentage of A, C, G and U of the sequences in it.
Additionally, your program should take as second argument an output file and document all sequences that are able to form perfect hairpin structures. Only valid transformed RNA sequences are considered for hairpin structures. We consider a sequence to form a hairpin structure if and only if the resulting RNA can be fold exactly in the middle of the sequence and we observe perfect complementary binding for each base. In case of odd sequence length, ignore the base in the middle of the sequence. To report the hairpin structure to the output file, only report the first half of the original RNA sequence in upper case letters. Do not report the full sequence. Do not report the header. For odd sequences length, do not report the unmatched base.

Sample output:

```
Lines: 40
Sequences: 20
GC content: 0.7
A: 0.1
C: 0.4
G: 0.3
U: 0.2
```

Hairpin sequences in input **FASTA**:

```
>seq1
AGCTAGCT
>seq2
aAGGCcCUU
```

Report Hairpin sequences:

```
AGCU
AAGG
```