Prof. Dr.-Ing. Ralf Schenkel
Tobias Zeimetz
Trier University

Wintersemester 21/22

Exercises for the Class

Elements of Computer Science: Programming

Assignment 05

Submission of solutions until 3:00 p.m. at 08.12.2021
at `moodle.uni-trier.de`

- Every task needs to be edited in a meaningful way!
- Please comment your solutions, so that we can easy understand your ideas!
- If you have questions about programming or the homeworks, just ask you teachers!

Exercise 1 (Evaluation: Numbers/Text)

Write (e.g. based on the previous assignment) a Java program that reads a `int` number $m > 1$ and determines how many prime numbers there are between 2 and m (both inclusive). (If $m = 1$ is entered, the program may again produce any output; the input and reading of m has already been performed in the specified program).

```
79
The number of prime numbers up to this input is 22
fast enough
```

Watch out: For this task, the computing time required for the evaluation is also measured. A fast solution requires less than one second per case for all evaluation examples. Slow solutions take much longer. The faster your program is, the more cases can be successfully evaluated within one second. For example, consider whether the prime number test for a given $n \leq m$ really needs to test all numbers between 2 and $n - 1$.

For slow but otherwise correct solutions there is a maximum of 10 points, the remaining 5 points are awarded on the basis of speed. The time measurement incl. the output **fast enough** resp. **too slow** is given in the program and must not be modified.

Advise: A nested loop solves the problem, but will unfortunately be too slow to get 15 points. Look at the *Sieve of Eratosthenes* for a quick solution.

Exercise 2 (Evaluation: Text)

Write a Java program that reads two numbers f and n as input and generates patterns (from $n \times n$ characters) of the following type as output (similar to example `K3B14E_Flag.java`). Here f specifies the format or the pattern of the output (here only 1 or 2, other values may be treated arbitrarily); $n \geq 5$ is always odd.

Format $f = 1$ as 7×7-Pattern:

```
1 7
.X.X.X.
X.X.X.X
.X.X.X.
X.X.X.X
.X.X.X.
X.X.X.X
.X.X.X.
```

Format $f = 2$ as 7×7-Pattern:

```
1 7
XXXXXXX
XX....X
X.X...X
X..X..X
X.X...X
XX....X
XXXXXXX
```

The patterns are to be created with nested `for` loops. `System.out.print` may only be used in the following three forms (but of course multiple times):

- `System.out.print(".")`,
- `System.out.print("X")` und
- `System.out.println()`.

So you have to build the output from single `.`, `X` and line separators. Also make sure that no line ends are missing and that you do not create unnecessary blank lines.

Exercise 3 (Evaluation: Text)

Write a program that reads a number n between 0 and 999 as `int` value from the scanner. Now convert this number into natural language and display it on the console.

Example:

```
 11 -> eleven
 17 -> seventeen
 34 -> thirty-four
100 -> one hundred
258 -> two hundred and fifty-eight
666 -> six hundred and sixty-six
812 -> eighth hundred and twelve
```

Negative inputs or inputs that are too large should return the output `Wrong Input!`:

```
Input: 258
In Words: two hundred and fifty-eight

Input: 1000
Wrong Input!
```

Watch out: The purpose of this task is not to program all numbers manually, but to find a minimal solution. For programs that have all numbers (or all numbers up to 100) programmed in, points will be deducted!