

## ASSIGNMENT (AFLEVERINGSOPGAVE) 4

### GENERAL INFORMATION

This assignment is made public on Friday, February 27th, 1 PM. The assignment is due on

Friday, March 5th, 1PM.

Hand in your assignment to the teaching assistant running your lab session.

The first page of your (written) assignment has to contain at least the following information:

- the course name (Grundlæggende Programmering)
- name and student number of the fellow student(s) in your group (max two)
- assignment number

*Please staple your assignment!*

You will get back the graded assignment one week after submission deadline.

### WARM-UP AND SUGGESTIONS FOR FURTHER EXERCISES

As warm-up I suggest that you have a look at the review questions of Chapter 4 of the course book. As to further exercises I suggest the following from Chapter 4 of the textbook (pp. 135–138 of my copy), where for good variation you should pick exercises from different items below.

- Exercises 3 and 4 are about implementing simple mathematical methods.
- Exercises 7 – 9 relate to previous examples of the book and require that you have studied those thoroughly.
- Exercise 12 asks you to implement a simple fortune telling program.
- Exercises 13 – 16 are about finding roots of mathematical functions and relate to Section 4.11 of the book.
- Exercise 17 is about a larger piece of code (again for a game) and should only be attempted if you feel really comfortable with the concepts so far.
- Exercise 20 is about *Fibonacci numbers*. To find out more about Fibonacci numbers check <http://www.mcs.surrey.ac.uk/Personal/R.Knott/Fibonacci/fib.html>.

It is up to you whether or not you want to work on these exercises. Those exercises will not be marked.

### ØVELSER

Arbejd i små grupper.

- Metoder: Opgaver 3, 4 fra kapitel 4.
- Metoder: Opgaven 12 fra kapitel 4.
- Rekursion: Opgave 20 fra kapitel 4.

## ASSIGNMENT: RECURSION

### AFLEVERINGSOPGAVE: REKURSION

Vi har en metode

- `String tail(String)`

hvor vi ved at metoden tager en tegnstreng og returnerer strengen uden det første bogstav. Eksempler er

- `tail("Hello World")`  
som returnerer strengen `ello World`,
- `tail("World")`  
som returnerer strengen `orld`,
- `tail("W")`  
som returnerer den tomme streng `ε`,
- `tail("")`  
som også returnerer den tomme streng.

En implementering af metoden, som vi ikke forventer at I forstår, er

```
static String tail(String s){return s.substring(1);}
```

Implementeringen kræver brug af pakken `lang`, dvs. I skal også have `import java.lang.*`; i jeres fil.

I Java kan man bruge metoden `length()` til at finde længden af en tegnstreng, f.ex. returnerer `"Hello World".length()` værdien 11. I denne opgave skal vi skrive to forskellige implementeringer af en metode `countChar(String)`, selvfølgelig uden at bruge metoden `length()`, som skal give de samme værdier som metoden `length()`, dvs. `countChar("Hello World")` vil (også) returnere værdien 11.

1. Brug løkker til at skrive en metode `int countChar(String)` der returnerer længden af en tegnstreng. Du må ikke bruge Javas metode `length()`, men kun metoden `tail()`. Løsningen kaldes en iterativ løsning.
2. Brug rekursion til at lave en ny implementering af metoden `int countChar(String)`.

Husk at man kan definere antallet af bogstaver i en tegnstreng som

$$a(s) = \begin{cases} 0, & \text{hvis } s \text{ er den tomme streng,} \\ 1 + a(t), & \text{hvis } s \text{ er ikke tom, og hvor vi får } t \\ & \text{ved at fjerne det første bogstav fra } s. \end{cases}$$

3. Test jeres metoder ved at skrive et program der spørger om input (en tegnstreng), og svarer med længden af input. Brug metoden `Console.in.readLine()` til at få en tegnstreng.

**Opgaven:** Aflever koden af de to metoder og kildekoden af jeres testprogram.