Exercises week 7

Preliminary version only first exercise

Last update: 2022/09/25

The goals of this week are to enable you to apply Java streams and parallelize Java streams and recognize possible applications of functional programming and lazy evaluation.

The goals are:

- Apply and examine a number of Java stream concepts including: stream sources, intermedialte operators and terminal operators
- Apply lambda expressions in Java.
- Apply benchmarking to the students own algorithms/methods written in Java

Not mandatory
If you are already comfortable with Java lambdas, you may skip this exercises. If you are not, please try to solve it and turn in your solution.

Exercise 7.1 You may use this Java skeleton as a starting point of the exercise.

```
import java.util.function.Function;
class LambdaExample {
  public static void main(String[] args) { new LambdaExample(); }

  public LambdaExample() {
    System.out.println("I: "+increment(f));
    //To be filled in
  }
  Function<Integer, Integer> f = (x) -> x+1;
}
```

You can find the code above in `Week07/code-exercises ... /LambdaExample.java`.

1. Write the (missing) code for the `increment` function to make the output of the `LambdaExample`: I: 9
2. Change the code in `LambdaExample` so that the function `f` multiplies with 5 (instead of incrementing).
3. These code snippets are from `Benchmark.java` and `Benchmarkable.java` in `Week05/code-exercises ... /...`:

```
---- Benchmark.java
import java.util.function.IntToDoubleFunction;
...
public Benchmark() {
...
    Mark6("multiply", i -> multiply(i));
    Mark6("multiply", Benchmark::multiply);
...
}

public static double Mark6(String msg, IntToDoubleFunction f) {
    ...
    dummy += f.applyAsDouble(i);
}
---- Benchmarkable.java
import java.util.function.IntToDoubleFunction;
```

```
public abstract class Benchmarkable implements IntToDoubleFunction {
  public void setup() { }
  public abstract double applyAsDouble(int i);
}
```

Write a short explanation of what happens in the two lines (emphasize explaining the two lambda expressions):

```
    Mark6("multiply", i -> multiply(i));
    Mark6("multiply", Benchmark::multiply);
```

4. Write a new version of `Mark6` called `Mark6int` that will *only* accept measuring functions that takes an integer argument and delivers an integer result (e.g. `intcountSequential` in Exercise 7.2). Like `Mark6`, `Mark6int` should measure the running time of the function given as the second argument.

```
    public static double Mark6int(String msg, ???) {
      //To be filled in
    }
```

Challenging

5. Can you write a method `void swap(int a, int b)` that will swap the values in `a` and `b`? Note the "?", so a possible solution to this exercise is no.

   Regardless of whether you answer yes or no, write a short justification of the answer.

   I strongly encourage you **not** to try to "Google" the answer. Try to come up with it yourself.