Programming in Java
Jacobs University Bremen
Dr. Jürgen Schönwälder

Course: 320341
Date: 2015-09-11
Due: 2015-09-18

Problem Sheet #1

Problem 1.1: *Java compiler and virtual machine options* (2+2 = 4 points)

What do the following options of the Java compiler (`javac`) do? Try them out on the `Factorizer.java` program.

- `-verbose`
- `-classpath`
- `-g`
- `-Xlint`

What do the following options of the Java virtual machine (`java`) do? Try them out on the `Factorizer.java` program.

- `-verbose`
- `-Xprof`
- `-Xint` and `-Xmixed` and `-Xbatch`

Problem 1.2: *collatz sequence* (2+2+2 = 6 points)

The so called Collatz sequence a_i (named after the mathematician Lothar Collatz) is defined as

$$a_i = \begin{cases} n & \text{if } i = 0 \\ f(a_{i-1}) & \text{if } i > 0 \end{cases}$$

with

$$f(n) = \begin{cases} n/2 & \text{if } n \equiv 0 \pmod 2 \\ 3n+1 & \text{if } n \equiv 1 \pmod 2 \end{cases}.$$

The Collatz conjecture, also known as the $3n+1$ conjecture, states that the Collatz sequence will always reach 1 for any natural number n.

Write a Java program that reads positive numbers from the standard input and prints the Collatz sequence a_i to the standard output until the sequence has reached 1. The program terminates when the end of the standard input has been reached. You program must implement the algorithm twice, using primitive 64-bit signed numbers and `BigInteger` numbers. A sample execution of the program is shown below (first line is user input, second line is program output and so on).

```
$ echo "11" | java Collatz
11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

Compare the performance of the Collatz algorithm implemented using primitive types with the performance of the Collatz algorithm implemented using `BigInteger` objects.