Comp 250: Assignment 2

Instructors: Mathieu Blanchette & Jérôme Waldispühl

Due on October 9, 2015 at 11:59pm

- Your solution must be returned electronically on MyCourse.
- The only format accepted for written answers is PDF. PDF files must open on SOCS computers. Any additional files (e.g. images must be included in the PDF). Do not submit a compressed repository including your files (e.g. rar or zip files). Upload instead each PDF or text file individually. Submissions that do not follow these guidelines will be rejected.
- The solution of programming questions must be written in java. Your program should compile and execute of SOCS computers. Java files that do not compile on SOCS computer will not be graded.
- To some extent, collaborations are allowed. These collaborations should not go as far as sharing code or giving away the answer. You must indicate on your assignments the names of the persons with who you collaborated or discussed your assignments (including members of the course staff). If you did not collaborate with anyone, you write "No collaborators" at the beginning of your document. Importantly, if asked, you should be able to orally explain your solution to a member of the course staff.
- Unless specified, all answers must be justified.
- When applicable, your pseudo-code should be commented and indented.
- Violation of these rules are subject to penalties.
- Partial answers will receive credits.
- The course staff will answer questions about the assignment during office hours or in the online forum at `https://cs250qanda.cs.mcgill.ca/`. We urge you to ask your questions as early as possible. Questions asked within 24 hours of the deadline may not receive a response in time.
- You will need a Java template file to answer Questions 3 and 4. You can download this file at `http://www.cs.mcgill.ca/~jeromew/COMP250material/HW2.java`

Question 1 **(Induction proof)** . . . . . (20 points)

Recall the definition of the Fibonacci numbers: $F(0) = 0$ and $F(1) = 1$ and $F(n) = F(n-1) + F(n-2)$ for $n \geq 2$. Prove by induction that:

(a) (10 points) for all $n \geq 1$, F(3n) is even.

(b) (10 points) for all $n \geq 0$, $F(n) = \frac{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n}{\sqrt{5}}$.

Question 2 **(Towers of Hanoi)** . . . . . (30 points)

The "Towers of Hanoi" problem is the following: You are given three vertical rods (the towers), numbered 1, 2, and 3. On the first rod are stacked n rings of decreasing size (see figure). The goal is to transfer all rings on the second rod. The only operation allowed is to take a single ring from the

top of one of the towers and to move it to the top of another tower, *provided the ring is not larger than the ring already on top of that tower (if any)*. In other words, for each towers and at any point in time, the rings have to be stacked in decreasing size.

The question is to write a recursive algorithm that prints the series of moves required to transfer all n rings from rod i to rod j. This seems awfully difficult until you realize the following. Assume rod i has the m smallest rings of all three towers. To transfer the m smallest rings from rod i to rod j, all one needs to do is to

1. first transfer the $m-1$ smallest rings from rod i to rod $k = 6 - i - j$,
2. then transfer the m-th ring from rod i to rod j,
3. finally re-transfer the $m-1$ smallest rings from rod k to rod j.

Notice that step (1) and step (3) are just smaller versions of the same original problem. See for example the procedure described in the figure for $m = 3, i = 1, j = 2$. The question is to write a procedure Hanoi(m, i, j) that prints (using the pseudocode keyword "print") the series of moves needed to move the top m rings from rod i to rod j. For example, Hanoi(3,1,2) should print:

```
Move top ring from rod 1 to rod 2
Move top ring from rod 1 to rod 3
Move top ring from rod 2 to rod 3
Move top ring from rod 1 to rod 2
Move top ring from rod 3 to rod 1
Move top ring from rod 3 to rod 2
Move top ring from rod 1 to rod 2
```

(a) (15 points) Complete the pseudocode below:
Algorithm $Hanoi(m, i, j)$
Input: The number m of rings to be moved from rod i to rod j. It is assumed that rod i has at least m rings and that these are the m smallest of all three rods.
Output: Prints the series of step required to move the m smallest rings from rod i to rod j.
if (. . .) then
    Hanoi(. . . , . . . , . . .)
    print "Move top ring from rod . . . to rod . . .
    Hanoi(. . . , . . . , . . .)

(b) (5 points) Let $T(m)$ be the number of moves required to move m rings from one rod to another using your algorithm. Give a recurrence expressing $T(m)$ in terms of $T(m-1)$.

(c) (10 points) Guess an explicit formula for $T(m)$ (i.e. a formula that doesn't have $T(m-1)$ on the right part) and prove by induction that your guess was correct for any $m \geq 0$.

Question 3 **(Factorials)** . . . . . (15 points)

(a) (10 points) Write the pseudo-code of an **iterative** algorithm to compute the value of $n!$ (Note: By definition $0! = 1$). Characterize the type of the input, output, as well as pre-conditions and post-conditions. Then, prove the correctness of your algorithm using a loop invariant.

(b) (5 points) Implement your algorithm in the method `factorial(int n)` in the file `HW2.java`.

(Note: `factorial` returns long instead of int in order to augment the precision and reach values up to 20!)

Question 4 **(Binomial coefficient)** . . . . . (35 points)

The binomial coefficient $\binom{n}{k}$ is the number of ways of picking k unordered outcomes from n possibilities. Its value can be calculated explicitly as: $\binom{n}{k} = \frac{n!}{(n-k)!\cdot k!}$

It is also possible to compute the binomial coefficients recursively using the following definition:

$$\binom{n}{k} = \begin{cases} 1 & \text{if } k = 0 \text{ or } k = n \\ \binom{n-1}{k} + \binom{n-1}{k-1} & \text{for } n, k \text{ such that } 1 \leq k \leq n-1 \\ 0 & \text{otherwise} \end{cases}$$

(a) (10 points) Write the pseudo-code of your recursive algorithm. Characterize the type of the input, output, as well as pre-conditions and post-conditions. Then, prove the correctness of your algorithm.

(b) (10 points) Complete code of the recursive method `BCoeffRec(int n, int k)` in `HW2.java` to compute the value of binomial coefficient $\binom{n}{k}$ using the recursive definition. Comment your code and in particular show explicitly where are the base case and inductive step.

(c) (5 points) Complete code of the method `BCoeffExp(int n, int k)` in `HW2.java` to compute the value binomial coefficient using the explicit formula. Here, you should call the method `factorial(int n)` for calculating factorials implemented in the previous question.

(d) Now, we will compare the running time of the methods `BCoeffRec` and `BCoeffExp`. For this purpose, you will use the method `TimeBCoeffRec(int n)` (resp. `TimeBCoeffExp(int n)`) in `HW2.java` that calls `BCoeffRec(int n, int k)` (resp. `BCoeffExp(int n, int k)`) and returns the time used to calculate all $\binom{n}{k}$ in microseconds.

i. (5 points) Complete the method `benchmark(int n)` in `HW2.java`. This method must print the execution time of `BCoeffRec(n,k)` and `BCoeffExp` for all values of k such that $0 \leq k \leq n$. The format of the output is such that each row must start with the values n and k between parenthesis and separated by a comma. Then, the row contains the execution time in microseconds of `BCoeffRec(n,k)` followed by the execution time of `BCoeffExp(n,k)` (in this precise order) separated by a space or tabulation. We show below an example of the output of `benchmark(3)` (Note: execution times may vary).

```
(3,0)    3        3
(3,1)    1        0
(3,2)    1        1
(3,3)    1        1
```

Run the method `benchmark` for values of n of 10 and 20 and answer to the following questions:

ii. (2 points) Which method is the fastest? Explain why in one or two sentences.

iii. (3 points) For a fixed value n, are all execution times similar? For which values of k are the execution times the longest (resp. shortest)? Explain why in one or two sentences.