Computer Science

COMPSCI 105 S1C - Assignment 2

Due Date: Friday 16 May 2014 at 4:30pm

80 marks in total = 8% of the final grade

Assessment

- Due: 16 May 2013 (4:30 pm)
- Worth: 8% of your final mark

Files

- A copy of this handout, as well as relevant files for each of the questions in this assignment can be obtained from the 105 assignments page on the web:

  `http://www.cs.auckland.ac.nz/compsci105s1c/assignments/`

- Using the web drop box (`https://adb.auckland.ac.nz/`), you may submit the following files for this assignment:

  - Question 1: `Josephus.java`
  - Question 2: `Calculator.java`

Aims of the assignment

- solving problems using Linked Lists, Stacks and Queues

Warning

- The work done on this assignment must be your own work. Think carefully about problems you come across, and try to solve them yourself before you ask anyone else for help. Under no circumstances should you use code written by another person in your assignment solution.
- Penalties for copying will be severe – to avoid being caught copying, don't do it.
- To ensure you are not identified as cheating you should follow these points:
  - Always do individual assignments by yourself.
  - Never loan your code to another person.
  - Never put your code in a public place (e.g., forum, your web site).
  - Never leave your PC without locking the screen (e.g., to get food, to have a drink, or to go to the toilet). You are responsible for the security of your account.
  - Never get code from a tutor (e.g., private tutors). Several tutors have been caught giving the same code to all their students.
  - Always reference the source for text you copy as part of the answer to an assignment.

Your name, UPI and other notes

- In *all questions* your UPI **must** appear somewhere in the output for the program.
- In this handout, the UPI `abcd001` has been used to illustrate this in the examples, however make sure that for the programs **you** submit, **your** UPI is included somewhere in the output.
- All files should also include your name and ID in a comment at the beginning of the file.
- All your files should be able to be compiled without requiring any editing.
- All your files should include adequate documentation - note that you are not required to produce full Javadoc comments.
- This assignment has a set of resource files (`A2Resource.zip`) that you may find useful for the assignment. You do not have to use all of the resource files, but you must NOT modify any of them, because the markers will use the unmodified versions of the resource files to mark your assignment. The resource files are NOT to be submitted.

1. Question one (25 Marks)

The Josephus problem is named after a historian of the first century, Flavius Josephus, who survived the Jewish-Roman war due to his quick thinking and mathematical talents. Legend has it that he was one of 39 Jewish rebels trapped by the Romans. His companions preferred suicide to capture, so they decided to form a circle and to kill every 7th person in the circle, proceeding in a clockwise direction (starting from the first position) until only one was left (who was then expected to kill himself). Josephus wasn't excited by the idea of killing himself, so he calculated where he has to stand (the 17th position) to survive the vicious circle.

You will solve a more general version of this problem by using a linked list to simulate the circle of rebels. The following implementation requires the input of how many people will stand around the circle and how many shall be passed over before the next one is eliminated. For this question, you need to use the Linked List structure to solve the problem, i.e. use the `ListReferenceBased` class to write your implementation of the Josephus problem. The Josephus program will take two parameters from the command line:

- an int, which is the total number of people in the circle
- an int, which is the steps taken before the next elimination

The program should output the elimination sequence and calculate the safest position. Your program (`Josephus.java`) should output similarly as follows:

```
C:\105\Assignment2\Q1>java Josephus 13 7
Josephus problem by abcd001:
Total number of people: 13,
Eliminate every 7th person.
Solution...
Remove person at position 7
Remove person at position 1
Remove person at position 9
Remove person at position 4
Remove person at position 13
Remove person at position 11
Remove person at position 10
Remove person at position 12
Remove person at position 3
Remove person at position 8
Remove person at position 2
Remove person at position 5
The safest position in the circle is 6!
```

NOTE:

- Your program must use the `ListReferenceBased` class to solve the problem.
- For this question, you should submit the `Josephus.java` class.
- Your UPI must appear in the output, as shown in the example above

2. Question two (50 Marks)

Design and implement a simple mathematical expression calculator using the Stack data structure. The operators include '+, -, *, /' and parentheses such as '(,)'; the operands consist of integer numbers (single or multi-digit). The mathematical expressions are in infix format, and operators such as '*, /' have higher precedence than that of operators '+, -'. Assuming there are no unary operators and no spaces in the expressions.

For this question, you need to write a program called `Calculator.java`, which

1) checks the syntax of the expression and converts the infix form into its equivalent postfix form (using space to separate the numbers and operators);
2) calculates the value of the mathematical expression in the postfix form. Your program must compute the value of the expression correctly.

The mathematical expression will be given in an infix form. Your program should print out its postfix expression and the calculated value. You are required to use the stack data structure to solve this problem. Your program (`Calculator.java`) should behave similarly as follows:

```
C:\105\Assignment2\Q2>java Calculator "(8+12)*3/(14-9)"
A simple calculator by abdc001:
Evaluating ...
Postfix form: 8 12 + 3 * 14 9 - /
Result: 12.0

C:\105\Assignment2\Q2>java Calculator "5+(2-4/5*6"
A simple calculator by abdc001:
Evaluating ...
Syntax error: brackets mismatch -> too many open brackets.
```

Hints:

- Use a Stack structure to convert the infix expression to its postfix form and then use a Stack structure to compute the value of the postfix expression (both algorithms are taught in the lecture).
- In the case of checking syntax errors of a mathematical expression, there may be six major types of errors that you could consider:
  - invalid inputs, e.g., non-digit character;
  - brackets mismatch, e.g., too many open brackets, too many closing brackets;
  - missing operand, e.g., two consecutive operators, '(' followed by operator, operator followed by ')';
  - missing operator, e.g., digit followed by '(', ')' followed by digit, ')' followed by '(';
  - division by zero, e.g., '/' followed by 0;

Notes:

- Your program must use the `StackReferenceBased` class to solve the problem.
- For this question, you should submit the `Calculator.java` class.
- Your UPI must appear in the output, as shown in the example above

SUMMARY OF SUBMISSION INSTRUCTIONS:

You should submit the following files for this assignment through the web drop box at (**`https://adb.auckland.ac.nz/`**).

- `Josephus.java`
- `Calculator.java`
- `A2.txt` – a text file containing your feedback on the assignment (see below).

```
CompSci 105 S1 2014
===================
Feedback on Assignment Two
--------------------------
Name:
Login:
ID:

All the work done in this assignment is my own work. (Yes/No)

How much time did the assignment take overall?

What areas of the assignment did you find easy?

What areas of the assignment did you find difficult?

Which topics of the course did the assignment most help you understand?

Any other comments:
```

Summary of Marking details

Your UPI must appear in the output for every question. If it does not appear in the output, you will forfeit the marks for that question. All files should include your name and ID in a comment at the beginning of the file.

1. Question one	***25 Marks***
Submit a file called `Josephus.java` (your program must use the Linked List data structure to solve the problem)	
Documentation/ Neatness/ Style	3
Correctly output the elimination sequence	15
Correctly output the safest position	7

2. Question two	***50 Marks***
Submit a file called `Calculator.java` (your program must use the Stack data structure to solve the problem)	
Documentation/ Neatness/ Style	3
Correctly checks the syntax of input mathematic expression	10
Correctly converts the infix form into its equivalent postfix form	22
Correctly compute the value of the postfix expression	15

3. Feedback file	***5 Marks***
Submit a file called `A2.txt,` containing your feedback on the assignment.	
Submit your completed feedback for asssignment 2 in `A2.txt`.	5