Faculty of Engineering, University of Jaffna

Department of Computer Engineering

EC5080: Software Construction

Lab 01: Introduction to Java language features

Question 01 and Question 02 should follow the good programming practice involving *Control constructs, static / dynamic typing, scope, and name-spaces* and will be considered when grading your program.

Question 01:

This question allows a user to enter an algebraic expression of arbitrary length and view the simplified result. Each algebraic expression is stored as a String object and passed to an ExpressionResult class constructor. The integer result is computed by the object and printed along with the original expression.

1. First, create an IntelliJ Java project and name it as EC5080_Lab1_RegNo
2. Starting at the topmost line of the file, insert the following under the comments. You need to choose one of them. If you get any assistance from anyone/book/internet please include that. (*This is to have good practice of writing bibliographical reference*).

Certificate of Authenticity: (choose one from below)

// I certify the code of this lab is entirely my own work.

(or)

// I certify the code of this lab is entirely my own work,

// but I received assistance from [insert name].

// Follow this with a description of the type of assistance.

3. In the main method,

    (a) Repeatedly prompt the user for an expression of the form supported by ExpressionResult (see below).

    (b) For each expression provided by the user, create an ExpressionResult reference variable and object.

    (c) Print the value returned by a call to the object's toString() method.

    (d) End the loops when the user enters an "X".

4. Write a new class named ExpressionResult in the same package as the class Main to model an algebraic expression with integer operands that stores an expression of the form:

operand operator operand[operator operand ...]

in a String field that can only be accessed by its identifier within the class, parses and evaluates the expression, then stores the result in an integer field with similar class-only access. The expression is assumed to have at least two operands (integers) and at least one operator, each separated by a single space. Additional operators and operands may also appear in pairs such as:

"1 + 2" , "10 + 5 * 2" , "-11 + 6 / 2" , "4 * 1 + -2 ^ 3"

All evaluations are conducted from left to right and all results consider only integer arithmetic. The valid operators are: + (addition), - (subtraction), * (multiplication), / (integer division), and ^ (exponentiation or power function).

The following members are implemented in the ExpressionResult class and **no others**.

(a) The class ExpressionResult has two private fields: a String named expression and an integer named result.

(b) The default constructor sends the String literal containing only the digit 0 (zero) to the parameterized constructor, defined next.

(c) The parameterized constructor accepts one String expression as a formal parameter, saves the parameter into the expression field, and sets the result field with the return value of evaluate given expression as the actual parameter.

(d) The public static method named evaluate returns an int and evaluates the String expression sent as a formal parameter. The returned value is the integer corresponding to the expression's simplified value.

Implementation details of the evaluate method follow:

i. Use the String.split method to parse the passed expression into a String array. The String-class method split accepts a String parameter which must be a String literal with a single space. The method will create a String array object with the expression's operands and operator(s) as elements.

ii. Iterate through the parsed String array elements in a loop to accumulate the integer result based on the operators and operands.

A. Set the initial value of result to the first (index 0) element by calling Integer.parseInt() with the first element (an operand) sent as the actual parameter.

B. Set the current operator based on the value of the array elements at an odd index.

C. Set the current operand to the array element's integer value when the index is any other value (i.e., not 0 and not odd).

D. Update the value of result by performing the operation indicated by the current operator using result and the current operand as the inputs.

iii. Finally, the result is returned upon loop completion.

The following example illustrates how an expression is evaluated. Given the expression 1 + 4 * 8, an array of Strings is created with the five elements "1", "+", "4", "*", and "8". Then:

• result is initialized to 1

• operator is set to "+"

• operand is set to 4 and result is updated to 1 + 4 = 5

• operator is set to "*"

• operand is set to 8 and result is updated to 5 * 8 = 40

(e) A public String method toString accepts no parameters and returns the expression and result fields formatted with an equals sign between them.

Some examples: "10 + 5 * 2 = 30", "4 * 1 + -2 ^ 3 = 8"

Question 02:

This question is to practice the different ways of *Garbage collection in Java*. You can use any IDE to work on this question.

1. Explain with examples, how you can make an object eligible (different ways) to garbage collect in Java?
2. Compile, run the program and answer the following questions. (You may need to write the extra lines of code to explain the output for each question.)

```
class Registration{

int regno; //Last two digits of your registration number

String name;


 public void setDetails(int a,String b){

  regno=a;
```

```
  name=b;
 }
 public void showDetails(){
  System.out.println("Registration number is "+regno);
  System.out.println("Name is = "+name);
 }
 public static void main(String args[]){
  Registration s1 = new Registration();
  Registration s2 = new Registration();
  s1.setDetails(55,"xxxx");
  s2.setDetails(21,"yyyy"); //Replace xxxx, yyyy with any names
  s1.showDetails();
  s2.showDetails();
  }
}
```

a. Two objects and two reference variables are created in the given program. Write the lines of code to create a third reference variable and point it to “s2”. Explain the output.

b. Set s2 to null. Explain the output.

c. Set s3 to null. Explain the output.

3. Compile and run the following code. And answer the question below.

```
public class Message {
String message;
public Message(String msg) {
this.message = msg;
}
public void display() {
print(this.message);
}
public void print(String message) {
```

```
Message msg = new Message("The message: " + message);
}
public static void main(String[] args) {
Message msg_1 = new Message("SC1");
Message msg_2 = new Message("SC2");
msg_1 = msg_2;
msg_1.display();
new Message("SC3").display();
msg_1 = null;
System.gc();
}
public void finalize() {
System.out.println("" + this.message + "" + " successfully garbage collected");
}
}
```

a. Paste the output here.
b. Explain the reason of the output by highlighting the ways the objects were garbage collected.

Create a zip file in a format of Regno-Coursecode including all your code folders and pdf answer sheets.

Upload the zip file on/before given deadline via team.

Any plagiarized work will be given 0 marks.