CSC 241, Data Structures and Algorithms Project 1

The program and its execution are due via D2L. Late programs are not accepted.

1. Implement the following polynomial function wit a single linked list.

```
public class PolyA extends SingleLinkedList<String>  {
…
public static void main(String args[]){
   PolyA a = new PolyA ();
   PolyA b = new PolyA ();
   PolyA c = new PolyA ();
   a.initialization();
   b.initialization();
   c = a.add(b);
   System.out.println(a);
   System.out.println(b);
   System.out.println(c);
 }
}
```

When the inputs for a and b are $3.2x^2 - 4x + 5$ and $-2x^7 + 4x^2 - 5$, c will be `-2.0X^7 + 7.2X^2 - 4.0X^1` That is, $-2x^7+7.2x^2 – 4x$. The SingleLinkedList class can be downloaded from www.cs.wcupa.edu/~zjiang/SingleLinkedList.java. We assume that the exponent is always non-negative, that is, 0, 1, 2, … but the coefficient can contain decimal part.

2. Implement the polynomial function with the required inheritance.

```
public class PolyB extends SingleLinkedList<polyUnit>  {
…
public static void main(String args[]){
   PolyB a = new PolyB ();
   PolyB b = new PolyB ();
   PolyB c = new PolyB ();
   a.initialization();
   b.initialization();
   c = a.add(b);
   System.out.println(a);
   System.out.println(b);
   System.out.println(c);
 }
}
```

The polyUnit class can be downloaded from
www.cs.wcupa.edu/~zjiang/polyUnit.java

3. Implement the polynomial function with the double cyclic linked list.

```
public class PolyD extends DoubleCyclicLinkedList<polyUnit>
{
…
public static void main(String args[]){
   PolyD a = new PolyD ();
   PolyD b = new PolyD ();
   PolyD c = new PolyD ();
   a.initialization();
   b.initialization();
   c = a.add(b);
   System.out.println(a);
   System.out.println(b);
   System.out.println(c);
 }
}
```

The DoubleCyclicLinkedList class can be downloaded from
www.cs.wcupa.edu/~zjiang/DoubleCyclicLinkedList.java

4. Implement the MyStack as required in the follows:

```
public class MyStack<E> extends SingleLinkedList<E>
implements StackInterface<E>{
…
}
```

The interface StackInterface can be downloaded from
www.cs.wcupa.edu/~zjiang/StackInterface.java

Use such a stack to support both the following tasks in one main method.

a. Check for balancing symbols (), {} and [] in a plain text file. You can test your program with the following cases (reference answer: T, F, F):
(a + b * { c / [d – e] }) * (d – [e – f])
(a + b * { c / [d – e }]) * (d – [e – f])
(a + b * { c / [d – e]] }) * (d – [e – f])

b. Given a given postfix expression, print out the result (the last double type number left in the stack). A sample is available here:
3 2 * 4 5 1 - / +

Evaluation:

1. The assignment is based on the materials in class. Please check the ppt file available at the class website and use the URL links to download the relevant programs/materials for study.
2. Submit all executable java files. The resulting display CANNOT contain any unnecessary 0 or unary plus operator+. Submit source code with the file name listed as required.
3. Do not forget to backup all your files in case your submission is lost. It will be your responsibility to resubmit the project if the instructor did not receive your submission.