Part A - coding

1. Write a method called isMatch that takes a string of parentheses and check if all parentheses match correctly. The parameter string might consist of other characters. Ignore other characters, just check the () {} []

```
   public static boolean isMatch(String expressionLine);
```

Use a JCF ArrayDeque or LinkedList as a stack to store all open or left parentheses.
Name your file as A2Match then follow by your initials at the end. Sample John Smith: A2MatchSJ.java
Use following main method to check your method.

```
public static void main(String[] args){
   String[] expression = new String[]{"{5*(x+2)+(y-1);}", "32*(20+(x[i]*3)-1",
"(){([][])}", "({}((0))", "{([]})", "{}())", "{"};
   for (String st: expression)
      System.out.println(st + " is " + isMatch(st));
}
```

2. Write a general Tree class method

```
public void levelOrder();
```

Which starts at the root of a tree and prints all elements of the tree in level-order (in one line).

It may call a private levelOrder method which takes a waiting queue of nodes to be printed. If so, please implement the following method as well.

```
private void levelOrder(List<TreeNode<E>>)
```

3. Write a general Tree class method

```
public void postOrder();
```

Which starts at the root of a tree and prints all elements of the tree in post-order (in one line).

It may call following private method which should also be implemented if called.

```
private postOrder(TreeNode<E>)
```

4. Write a general Tree class method

```
public int height(TreeNode<E>);
```

Which takes a node as its parameter and check the height of the given node. This method is called in the following method.

```
public int height(){
   return height(root);
}
```

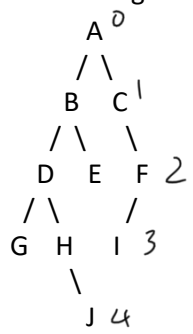
Extra Credit. Write a general Tree class method

```
public void isSubTree(TreeNode);
```

Which takes a node of a subtree, and check whether the subtree starting at the parameter node is a subtree of current tree.

Part B – written (Submit as an pdf file, or hand-in in class.)

5. Consider following tree



a. What is the height of the tree?

4

b. What is the height of node E?

0

c. What is the output of pre-order traversal?

ABDGHJECFI

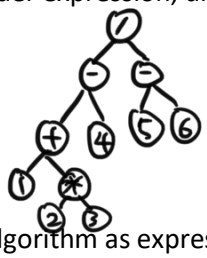
d. What is the depth of the node E?

2

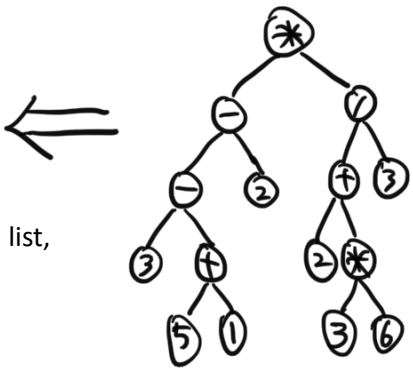
e. Is E an internal node?

No

6. Consider following post-order expression, draw the expression tree.
123*+4-56-/



7. Draw the following math algorithm as expression tree.
(3-(5+1) – 2) * (2+3*6) / 3



8. Consider following function, it should remove even values in list,

```
public static void removeEven(List<Integer> list){
  for (int i : list){
    if (i % 2 == 0) list.remove(i);
  }
}
```

Does this one work? If not, write the correct version.

```
Public static void removeEven(List<Integer>list){
    for(int i=0; i<list.size();i++){
      if(list.get(i)%2==0){
        list.remove(i);
    }
  }
}
```