

CS 102: Practice Test Problems for the Material after Test 2

1. Consider G_n defined by

$$G_n = \begin{cases} 0 & \text{if } n = 0 \\ 3 & \text{if } n = 1 \\ 2 + G_{n-1}G_{n-2} & \text{if } n > 1 \end{cases}$$

- (a) Compute G_4 . Show your work.
 - (b) Write a recursive method whose return value is G_n .
 - (c) Write an iterative method whose return value is G_n .
2. For `LStack` is an implementation of the `Stack` interface which stores a stack of integers, what is the output to the screen of the following code fragment?

```
Stack s = new LStack();
s.push(5);
s.push(10);
s.push(3);
System.out.println(s.pop());
int v = s.pop()*s.pop();
s.push(v);
s.push(13);
int w = s.pop();
int x = s.pop();
s.push(x-w);
System.out.println(s.pop());
```

3. Write a method

```
public void resize()
```

which takes an array `ar` of `Objects` (implemented as global instance data), replaces `ar` with an array of twice the size of the original `ar`, and copies all the original values of `ar` into the first slots of the new `ar`.

4. What is the output to the screen for the following Towers of Hanoi program? Note the main method is inside the `TowersOfHanoi` class, but the rest of the code is the same as we discussed in class.

```
public class TowersOfHanoi{
    private int totalDisks;
    public static void main(String[] args){
        TowersOfHanoi towers = new TowersOfHanoi(3);
        towers.solve();
    }
    public TowersOfHanoi (int disks){ totalDisks = disks; }
    public void solve(){
        moveTower(totalDisks,1,3,2);
    }
    private void moveTower(int numDisks, int start, int end, int temp){
        if (numDisks == 1) moveOneDisk(start,end);
        else{
            moveTower(numDisks-1, start, temp, end);
            moveOneDisk(start, end);
            moveTower(numDisks-1, temp, end, start);
        }
    }
    private void moveOneDisk(int start, int end){
        System.out.println("Move one disk from " + start + " to " + end);
    }
}
```

5. Consider the function $f(x) = e^x - x - 2$. Note that $f(0) = -1$ and $f(2) = e^2 - 3 > 0$. Assume $f(x)$ is implemented in a method

```
public double f(double x){ return Math.exp(x) - x - 2; }
```

Implement a binary search algorithm which returns a root of $f(x)$ in the interval $(0, 2)$, within a tolerance of $1E-10$.

6. Recall that a queue has two basic operations, `enqueue` and `dequeue`. Assume `q` is a queue which holds `Objects`. Write code which sends the `Object` at the front `q` to the end of `q`.

7. Fill in the following code for a linked list of integers:

```
public class MyList{
    private Node head;

    public MyList(){ // constructs an empty list: fill in code:

    }

    public void insertAtHead(int n){
        // inserts new node containing n at head of list: fill in code:

    }

    public void insertAtIndex(int index, int n){
        // inserts new node at index i: fill in code:

    }

    private class Node{
        public int num;
        public Node next;
        public Node(int n){
            num = n;
            next = null;
        }
    }
}
```

8. Consider Postfix.java as discussed in class:

```
import java.util.Scanner;
public class Postfix{
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter a postfix expression:");
        String st = scan.nextLine();
        Stack s = new LStack();
        Scanner stringScan = new Scanner(st);
        while (stringScan.hasNext()){
            String token = stringScan.next();
            if (token.equals("+") || token.equals("-")
                || token.equals("*") || token.equals("/")
                || token.equals("%")){
                //binary operation code
                int operand2 = s.pop();
                int operand1 = s.pop();
                int val;
                if (token.equals("+"))
                    val = operand1 + operand2;
                else if (token.equals("-"))
                    val = operand1 - operand2;
                else if (token.equals("*"))
                    val = operand1 * operand2;
                else if (token.equals("/"))
                    val = operand1 / operand2;
                else // token is "%"
                    val = operand1 % operand2;
                s.push(val);
            }
            else{ // assume number
                int val = Integer.parseInt(token);
                s.push(val);
            } // end else
        } // end while
        System.out.println("The evaluation of this postfix"
            + " expression is " + s.pop());
    } // end main
} // end Postfix class
```

Trace through the value of the stack if the following postfix string expression is entered by the user:

```
"16 3 - 7 10 5 / + 2 + %"
```

What is the evaluation of this postfix expression?

9. List all the elements (from head to tail) of an initially empty queue `q` of integers after the following code is performed:

```
q.enqueue(7);  
q.enqueue(9);  
q.enqueue(4);  
q.enqueue(q.dequeue() + q.dequeue());  
q.enqueue(15);  
q.dequeue();  
q.enqueue(12);
```