Name: _______________________________ **ID: A**

CIS2168 Fall 2014 Midterm

Short Answer (5 points each)

1. In the Array implementation of Queue, what does the folloing code do?

   q = (E[]) new Object[10];
   Answer:

   .

2. Assume the Stack size is fixed and the size is provided by the user when the Stack is instantiated. Member variable "CAPACITY" represents the size of the stack, while N represents the number of items in the stack. Write a method "booean isFull()", which returns true if the stack is full. It returns false otherwise.

   Answer:

   .

3. What does the following code fragment do to the queue q?

```
    Stack<String> s = new Stack<String>();
     while(!q.isEmpty())
       s.push(q.dequeue());
     while(!s.isEmpty())
        q.enqueue(s.pop());
```

   Answer:

4.
```
root |--100 -- |--a.txt
     |         |--- b.txt
     |
     | --200 --|--2001--|--c.txt
     |         |-- d.txt
     |
     |
     |---300
```

In folder roo, we have folders 100,200,2001,300 subfolder and other files. What is the output of following code?

```
Queue<File> q = new Queue<File>();
      File root = new File(root);
      q.enqueue(root);
      while (!q.isEmpty()) {
         File directory = q.dequeue();
         File[] files = directory.listFiles();
         for (int i = 0; i < files.length; i++) {
            if (files[i].isDirectory()) q.enqueue(files[i]);
            else System.out.println(files[i]);
         }
      }
```

Answer:

.

5. What does “Comparable” mean in the following class definition.

```
public class SortedBag<E extends Comparable<E>>
        extends Bag<E>
{
...
}
```

Answer:

.

6. What is the big O of the following method?

```
void foo(int n)
{
        while(n>1)
        {
                n = n/2;
                print(“hell”);
        }
}
```

Answer:

.

Problem

7. Write a method “int count(Node head)”, which returns the number of nodes in the linked list referenced by “head”.(10 points)

   Answer:

   .

8. Write a recursive method “void print(Node head”) to print all node data from head to tail order. (10 points)

   Answer:

   .

9. We want to implement this Queue interface. Items in this queue cannot be removed.

```
public interface Queue<E extends Comparable<E>> extends Iterable<E> {
    public void enqueue(E item);
    public E peek();
    public int size();
    public E min();
    public boolean isEmpty();
}
```

The public method min returns the minimum item in the queue. Describe a constant time (O(1)) algorithm for the “min” method and implement it. (10 points)

Answer:

.

10. Give the value of ex(6): (10 points)

```
    public static String foo(int n) {
        if (n <= 0) return "";
        return ex(n-3) + n + ex(n-2) + n;
    }
```

11. The LinkedBag<E> class uses the Node class and has the data fields

private Node head; // reference to head node of list representing this bag

private int N; // number of elements in this bag

Node class has data fields

private E data; //data field

private Node next; //references the next node in the list

(a) Write the toString method with prototype

public String toString()

that returns a string of the form [e1,e2,...,en] (10 points)

Answer:

.

(b) Write the add method with prototype

public void add(E element)

that adds the given element to this bag. (10 points)

Answer:

.

(c) Write the countOccurrences method with prototype

public int countOccurrences(E target)

that returns the number of times the given target element appears in this bag. Assume that null data elements are not allowed in the bag. (10 points)

Anwwer:

.

CIS2168 Fall 2014 Midterm
Answer Section

SHORT ANSWER

1. ANS:
creates an array of size 10 to hold the items in the queue

PTS: 1

2. ANS:

```
boolean isFull(){
{
        return N == CAPACITY;
}
```

PTS: 1

3. ANS:
Reverses the items on the queue.

PTS: 1

4. ANS:
a.txt,b.txt,d.txt, c.txt

PTS: 1

5. ANS:
items in the bag can be compared to one another.

PTS: 1

6. ANS:
O(logn)

PTS: 1

PROBLEM

7. ANS:

```
int count(Node head){
        int c = 0;
        whiel(head != null){
                c++;
                head = head.next;
        }
        return c;
}
```

PTS: 1

8. ANS:

```
void print(Node head){
        if(head== null) return;

        System.out.print(node.data);
        print(head.next);
}
```

PTS: 1

9. ANS:

keep a variable always points to the min items. Every time a new item is inserted, update min if necessary.

```
In the class:
private E minValue;

Modify enqueue(E item)
{
...

if(item.compareTo(minValue) < 0){ minValue = item}

}

/*
*       returns the minimum item in the queue
*/
piublic E min(){return minValue;}
```

PTS: 1

10. ANS:

```
ex(3)+6+ex(4)+6
ex(3) = ex(0)+3+ex(1)+3
ex(4) = ex(1) + 4 + ex2(2) + 4
ex(1) = 11


311361142246
ex(3) = 3113
ex(4) = 114224
ex(1) = 11
ex(2) = 22

311361142246
```

PTS: 1

11. ANS:

(A)

```
public String toString()
{
  Stirng s = “[”;
  Node current = head;
  while(current != null){
         s += current.data + “,”;
  }
  s += “]”
  return s;
}
//if students do not have “[“ and “]”, it is ok.
```

(B)

```
public void add(E element){
         Node t = new Node();
         t.data = element;
         t.next = head;
         head = t;
         N++;
}
```

Or add the new node to tail

```
public void add(E element){
         Node t = new Node();
         t.data = element;
         Node tail = head;
         while(tail.next != null) tail = tail.next;
         tail.next = t;
         N++;
}
```

(C)

```
public int countOccurences(E target){
         int c = 0;
         Node current = head;
         while(current != null){
                  if(current.data.equals(target)) c++;
                  current = current.next;
         }
         return c;
}
```

PTS: 1