CSC 212 Midterm 1 Solution - Fall 2013

Exam Duration: 2 Hours

13/11/2013

Question 1 [25 points]

1.

	Statement	S/E	Frequency	Total
1	public static int getNum(int n)	0	-	0
2	{	0	-	0
3	n=1;	1	1	1
4	while(n <5) {	1	5	5
5	System.out.println(”Less than 5”);	1	4	4
6	n++;	1	4	4
7	}	0	-	0
8	return n;	1	1	1
9	}	0	-	0
	Total operations			15
	Big-oh			$O(1)$

2.

	Statement	S/E	Frequency	Total
1	void addThemUp(int n) {	0	-	0
2	int sum=0;	1	1	1
3	for(int i=0; i<n; i++) {	1	$n+1$	$n+1$
4	for(int j=1; j<=10; j++) {	1	$11n$	$11n$
5	sum= sum+i+j;	1	$10n$	$10n$
6	System.out.println(sum);	1	$10n$	$10n$
7	}	0	-	0
8	}	0	-	0
9	System.out.println(”Good bye”);	1	1	1
10	}	0	-	0
	Total operations			$32n+3$
	Big-oh			$O(n)$

. . . .

Question 2 [25 points]

1. See slides.
2.

```
public volid removeTail(){

        tail= tail-1;
        if(tail<0)
                tail= maxSize-1;
        size--;
}

public volid removeHead(){

        head= (head+1)%maxSize;
        size--;
}
```

Question 3 [25 points]

1.

```
public static Integer findSmallest(Queue<Integer> q) {

        Integer min= q.serve();
        q.enqueue(min)
        for(int i=0; i<q.length()-1; i++){
                Integer tmp= q.serve();
                q.enqueue(tmp);
                if(tmp<min)
                        min= tmp;
        }

        return min;
}
```

2.

```
public static <T> void exchange(Stack<T> st1, Stack<T> st2) {

        Stack<T> st3= new Stack<T>();
        Stack<T> st4= new Stack<T>();

        while(!st1.empty())
                st3.push(st1.pop());

        while(!st2.empty())
                st4.push(st2.pop());

        while(!st4.empty())
                st1.push(st4.pop());

        while(!st3.empty())
                st2.push(st3.pop());

}
```

Question 4 [25 points]

1.

```
public static List<T> firstHalf(List<T> l) {
        List<T> l2 = new List<T>();
        int size = 0;

        if(!l.empty()) {
                l.findFirst();
                while(!l.last()) {
                        size++;
                        l.findNext();
                }
                size++;

                l.findFirst();
                for(int i = 0; i < size / 2; i++) {
                        l2.insert(l.retrieve());
                        l.findNext();
                }
        }

        return l2;
}
```

2.

```
public void removeDuplicates() {
        Node<T> cur1 = head;

        while(cur1 != null) {
                Node<T> prev = cur1;
                Node<T> cur2 = cur1.next;
                while(cur2 != null) {
                        if(cur1.data.equals(cur2.data))
                                prev.next = cur2.next;
                        else
                                prev = cur2;

                        cur2 = cur2.next;
                }
                cur1 = cur1.next;
        }
}
```

. . . .