Q1-1

1- Answer: (b)

2- Answer: (b) O(n log(n))

3- Answer: (e) O(n)

4-Answer: (c) O(1)

5-Answer: (c) O(1) – insert when curren is last, no loop needed.

6-Answer: (c) O(1)

Q1-2

	Statmenet	S/E	Freq	Total	Answer
1	int sum = 0;	1	1	1	b
2	for (int i = 0 ; i < n * n ; i++)	1	n^2+1	n^2+1	e
3	for (int j = n ; j < 2 * n ; j++)	1	$n^2(n+1)$	n^3+n^2	d
4	Sum += j	1	n^3	n^3	e
5	return sum	1	1	1	d
Total O				$O(n^3)$	c

Q2-1

```
public static <T> void removeDuplicate(LinkList<T> l, T k)
{
        l.findfirst();
        boolean found = false;

        while(! l.last() && ! found)
        {
                if (l.retrieve().equals(k))
                        found = true;
                else
                        l.findnext();
        }

        if (l.last() && l.equals(k))
                found = true;

        if (found)
        {
                l.findnext();

                do
                {
                        if (! l.last())
                        {
                                if (l.retrieve().equals(k))
                                        l.remove();
                                else
                                        l.findnext();
                        }
                }while(! l.last());

                if(l.last() && l.retrieve().equals(k))
                        l.remove();
        }
}
```

Q3:

```
public void insertAll(T e[],int n)
{
    int i = 0, k;

    while(i < n && size < maxsize)
    {
        for (k = size-1 ; k > current ; --k)
            nodes[ k + 1 ] = nodes[ k ];

        current++;
        nodes[current] = e[i];
        size++;

        i++;
    }
}
```

Using methods

```
public void insertAllM(T e[],int n)
{
    int i = 0;

    while(i < n && size < maxsize)
    {
        insert(e[i]);
        i++;
    }
}
```

User methods

```
    public static <T> void insertAllUser(ArrayList<T> a,T
e[],int n)
    {
        int i = 0;

        while(i < n && ! a.full())
        {
            a.insert(e[i]);
            i++;
        }
    }
```