Question 1

A) O(2^n)

B) O(n^2)

C) O(n^2)

D) O(n^2)

E) O(n)

Question 2

1)
```
public Student(){
    sno = 0;
    sname = "";
}
```

Question 2

1)
```
public Student(){
    sno = 0;
    sname = "";
}
```

Question 2

2)

```
public String getSname() {return sName;}
public void setSname(String _name) {sName = _name;}
```

Question 3

1)

```
public void addNoDuplicate(int element){
        for(int i=0;i<num;i++)
                if(data[i]==element) return;
                //extend space, copy old content
                if (num == data.length){
                        int biggerArray[] = new int[num*2 + 1];
                        for(int i=0;i<num;i++) biggerArray [i] = data[i];
                                data = biggerArray;
                        }
                        data[num] = element;
                        num++;
                }
}
```

Question 3

2)

No.

This method will also return true if bag.data is a subset of this.data.

E.g., given this.data=[1,2,3,4] and bag.data=[1,2], this function will return false.

Need to test whether these two bags sizes are the same.

Add

if(size()!=bag.size()) return false;

after

IntArrayBag bag = (IntArrayBag)o;

Question 4

1) Implement a method to remove an element e after the current node

```
public void removeNodeAfter (int e)
{
        IntNode preCursor = this;
        IntNode cursor = this.link;
        while(cursor!=null){
                if(cursor.data==e){
                        preCursor.link = cursor.link;
                        break;
                }
                preCursor = cursor;
                cursor = cursor.getLink();
        }
}
```

What's the time complexity of this algorithm? ___ O(n)_____

Question 4

2) For the above method, what's the time complexity?

First, define n=_____number of nodes in the linked list

Then, the complexity in Big-O is _____O(n)____

This function calculates the (sum of odd number)/(sum of even numbers)

Question 4

3) This function does addToEnd()

1->7->8->3->1->2

Question 4

4) This function gets the sublist starting from position x.

The first position node is at position 1.

4->4->2

Question 5

1)

```
public DNode getHead(){
        if(head.next==tail) return null;
        else return head.next;
}
```

Question 5

2)

```
public int listLength(){
        int ans=0;
        DNode<E> cursor = head.next;
        while(cursor!=tail){
                ans++;
                cursor = cursor.next;
        }
        return ans;
}
```

Question 5

10<->8<->10<->5