Question 0. [2 marks]
Complete the “identification section” at the top of page 1, then write your student number legibly at the bottom of every page of this test except page 1 (where indicated).

Question 1. [12 marks]
Complete the body of the method deleteFirst() in class StackUtils below according to its external and internal comments.

```java
public interface Stack {
    Object pop();
    void push(Object o);
    int size();
    boolean isFull();
}

public class S implements Stack { /* body not shown */ }

public class StackUtils {
    /** Remove the first item in the stack s (i.e. nearest the top) for
    * which v.equals(item) is true.
    * Requires: s and v are not null.
    * Ensures: The size of stack s is reduced by at most one item, and all the remaining
    * items in s are stored in the same LIFO order as the original stack s.
    * Returns: true only if an item equal to v was found and deleted. */
    public boolean deleteFirst(Stack s, Object v) {
        Stack temp = new S(Math.max(1, s.size()));
        Object item;
        boolean found;
        // Complete the method below. Do not declare any additional variables.

        while(s.size()>0 && !found) {
            item = s.pop();

            if (v.equals(item))
                found = true;
            else
                temp.push(item);
        }

        while (temp.size()>0)
            s.push(temp.pop());

        return found;

        // Continue on the back of this page if you need more room...
        } // end of removeFirst
    } // end of class StackUtils
```
Question 2.  [14 MARKS]

```java
public class P {
    public int i;
    public static int j;
    public P(int i) { this.i = i; }
    public void m(int j) {
        i = j;
        s(i);
    }
    public static void s(int i) {
        j = j + 1000;
        System.out.println("P: " + i + ", " + j);
        // DRAW THE MEMORY MODEL WHEN
        // YOU GET HERE FOR THE LAST
        // TIME.
    }
}
public class C extends P {
    public int i;
    public C(int i) {
        super(i);
        this.i = i + 100;
    }
    public void m(int j) {
        i = i + j;
        s(i);
    }
    public static void s(int i) {
        j = j + 1;
        System.out.println("C: " + i + ", " + j);
    }
    public static void main(String[] ar) {
        P p = new C(1);
        p.m(20);
        P q = new P(3);
        q.s(4);
    }
}
```

Part (a)  [12 MARKS]

Draw the memory model for this program the last time the method s(int) in class P is executed (possibly the only time), at the point indicated by the comment “DRAW THE MEMORY MODEL ...”. You only need to draw boxes for classes C and P in the static and object spaces (i.e. you can ignore Object, String, String[], System, and so on). Draw everything that is on the runtime stack at this point in the program execution.
Part (b) [2 MARKS]
What does the program print?

P: 121, 1000
P: 4, 2000
Question 3.  [12 MARKS]
The class `LinkedIntList` provides a linked list of `IntNodes`. Write the body of the method `deleteSmaller()` in this class which deletes each element of the list whose value is smaller than the maximum value of all the previous elements in the list. For example, if the original list contains elements 1, 3, 5, 2, 4, 5, 7 (in head to tail order), then the processed list contains the elements 1, 3, 5, 5, 7 (in the same order).

```java
public class IntNode { // Nodes for our linked list.
    public int value;
    public IntNode link;
}

public class LinkedList {  // Constructor, etc. not shown.
    private IntNode head;
    //** Delete each item in this list whose value is smaller than that of an
    // item which comes earlier in this list (i.e. closer to the head).
    // Ensures: The values in the new list increase, or stay the same, as
    // we iterate through the list from head to tail. (See example above.) */
    public void deleteSmaller() {

        if (head != null) {

            IntNode prev = head;
            IntNode curr = prev.link;

            while (curr != null) {

                if (prev.data > curr.data) {

                    // delete current node
                    prev.link = curr.link;
                    curr = curr.link;

                } else {

                    // step over current node
                    prev = curr;
                    curr = curr.link;

                }

            }

        }

    }

    // Continue on the back of this page if you need more room...

    // end of deleteSmaller()
}
// end of class LinkedList
```

Total Marks = 40