

QUIZ 1

Introduction to Computer Science (COMP 250)

Fri. Feb. 11, 2011

Professor Michael Langer

STUDENT NAME: _____ ID: _____

The exam consists of four questions, for a total of 10 points.
You may use the back of the sheet(s), if necessary.

No electronic devices (calculators) or notes are allowed.

1. **(2 points)**

(a) Convert 211 from decimal to binary.

(b) Convert 1101.101 from binary to decimal.

2. (2 points)

Consider the following sequence of stack commands:

`push(a)`, `push(b)`, `push(c)`, `pop()`, `push(d)`, `push(e)`, `pop()`, `pop()`, `pop()`, `pop()`.

- (a) What is the order in which the elements are popped ? (Give a list and indicate which was popped first.)

- (b) Change the position of the `pop()` commands in the above sequence so that the items are popped in the following order: **b,d,c,a,e**.

You are *not* allowed to change the ordering of the `push` commands.

3. (3 points)

Consider the following Java code for a singly linked list method `getIndexOf(E)` that returns the index of the first occurrence of given element in a singly linked list if that element is present, and it returns -1 if the element is not present. For example, if the given argument `e` is at the front of the list then the method returns index 0.

Provide the missing Java code of this method below.

```
public int getIndexOf(E e){ // You may assume the list is not empty,
                           // and head is defined in the usual way.

    SNode<E> cur = head;
    int index = 0;

    // ADD YOUR CODE HERE (AND ONLY HERE)

    if (cur.getElement() == e)
        return index;
    else
        return -1;
}
```

4. (3 points)

Suppose that you are given an array of n different numbers that increase from index 0 to index m , and decrease from index m to index $n - 1$. Note that there is a unique largest number in such a list, and it is at index m where $0 \leq m < n$. Here are a few examples.

	m	n
[3, 5, 17, 18, 21, 6]	4	6
[-3, 5]	1	2
[12, 7, 4, 2, -5]	0	5

Provide the missing code below of a recursive algorithm that returns the index m of the largest number, in time proportional to $\log n$.

The algorithm is initially called with `low = 0`, `high = n-1`.

In this question we do not require that your algorithm is written in Java.

```
findM( a, low, high ){ // array is a[ ], assume low <= high
    if (low == high)
        return low
    else{

        // ADD YOUR CODE HERE (AND ONLY HERE)

    }
}
```

Solutions

1.

- (a) 211 in binary is 1010011
- (b) 1101.101 in decimal is 13.625.

Marking scheme

1 point for each of (a) and (b). For (b), you got only 0.5 if the fractional part was left as a sum of powers of 2, namely $1/2 + 1/8 = 5/8$.

2.

- (a) c (popped first), e, d, b, a
- (b) push(a), push(b), pop(), push(c), push(d), pop(), pop(), pop() push(e), pop()

Marking scheme

1 point for each.

3.

```
public int getIndexOf(E e){

    SNode<E> cur = head;
    int index = 0;

    // BEGIN SOLUTION

    while ((cur.getElement() != e) && (cur.getNext() != null)){
        cur = cur.getNext();
        index++;
    }

    // END SOLUTION

    if (cur.getElement() == e)
        return index;
    else
        return -1;
}
```

Marking scheme

1 point for stopping condition that you find the element. 1 point for stopping condition at the end of the list. 0.5 points for each of the updates (`cur` advance and `index` increment)

4.

```
findM( a, low, high ){ // array is a[ ], assume low <= high
    if (low == high)
        return low
    else{

// ----- BEGIN SOLUTION 1 -----

        if (high-low == 1){
            if (a[low] < a[high])
                return high
            else
                return low
        }
        else{
            mid = (low + high)/2
            if (a[mid-1] < a[mid]){
                return findM(a, mid, high)
            }
            else
                return findM(a, low, mid)
        }

// ----- END SOLUTION 1 -----

// ----- BEGIN SOLUTION 2 (ALTERNATIVE) -----

        mid = (low + high)/2
        if (a[mid] < a[mid+1]){
            return findM(a, mid+1, high)
        }
        else
            return findM(a, low, mid)

// ----- END SOLUTION 1 -----
    }
}
```

Marking scheme

1 point for mid calculation. 1 point for correct if test. 1 point for correct recursive calls. (Only 0.5 if you forgot to return the value).