QUIZ 1

Introduction to Computer Science (COMP 250)

Mon. Feb. 2, 2009

Professor Michael Langer

STUDENT NAME: _____ ID: _____

The exam consists of four questions. There are a total of 10 points.
You may use the back of the sheet, if necessary.

No electronic devices (calculators) or notes are allowed.

1. **(3 points)**

   (a) Convert 34 to binary.

   (b) Verify your answer in (a) by converting from binary to decimal.

   (c) Convert 101.101 from binary to decimal.

SOLUTION:

(a) Following the algorithm given in class, you get 100010.

(b) $2^5 + 2^1 = 34$

(c) Converting 101 to decimal gives 5. Converting .101 to decimal gives

$$2^{-1} + 2^{-3} = .5 + .125 = .625$$

and so, adding the two decimal values together give the result: 5.625.
Some students wrote $5\frac{5}{8}$. We did not take off points for this.

2. **(2 points)**

What is the output of the code below, when you run class `C` ?

```
public class A {
   String s;
   public A(){
     s = "A default string";
   }
   public String toString(){
     return "toString from A: " + s;
   }
}

public class B extends A {
   public String toString(){
     return "toString from B : " + s;
   }
}

public class C {
   public static void main(String[] args){
     B   b  =  new B();
     A   a  =  b;
     System.out.println( a.toString() );
     System.out.println( b.toString() );
   }
}
```

SOLUTION:

```
toString from B:  A default string
toString from B:  A default string
```

3. **(2 points)**

Recall the insertion-sort method, which sorts an array '`double[] entries`' from largest to smallest:

```
public void insertionSort(){
  int cur;
  double tmp;
  for (int i=1; i < numEntries; i++){
     tmp = entries[i];
     cur = i;
     while ((cur > 0) && (tmp > entries[cur - 1])){
       entries[cur] = entries[cur - 1];
       cur--;
     }
     entries[cur] = tmp;
  }
}
```

Show the contents of the array below, after each pass through the `for` *loop.*

You will get 0 points for merely writing the numbers in their correctly sorted order.

```
index   entries    after 1    after 2
-----   -------    -------    -------    .....
  0      3.2
  1      4.1
  2     -1.0
  3      6.0
  4      5.6
```

SOLUTION:

```
index   entries    after 1    after 2    after 3    after 4
-----   -------    -------    -------    -------    -------
  0      3.2        4.1        4.1        6.0        6.0
  1      4.1        3.2        3.2        4.1        5.6
  2     -1.0       -1.0       -1.0        3.2        4.1
  3      6.0        6.0        6.0       -1.0        3.2
  4      5.6        5.6        5.6        5.6       -1.0
```

4. **(3 points)**

In lecture 9, you saw a Java method '`DNode remove(DNode node)`' for removing a node from a doubly linked list. Recall this method was defined in a class `DLinkedList`, which had fields `header` and `tailer`.

Fill in the missing code below for a second (overloaded) `remove` method that takes as input a positive integer `n`, removes the `n`-th node from the list, and returns a `boolean` which indicates whether the remove was successful or not. In particular, `remove(n)` returns `true` if the list has at least `n` elements and it returns `false` otherwise.

Use the extra page provided to sketch out your ideas, before writing your answer.

```
public boolean remove(int n){
    if (n <= 0)  return false;
    DNode cur = header;
    int i = 0;
    while (cur.next != tailer){

      i++;                    //   SOLUTION: ADD THIS LINE
      cur = cur.next;         //   SOLUTION: ADD THIS LINE
      if (i == n){            //   SOLUTION: ADD THIS LINE

        this.remove(cur);

        return true;          //   SOLUTION: ADD THIS LINE
        }                     //   SOLUTION: ADD THIS LINE
    }
    return false;
}
```