Introduction to Computer Science II (CSI 1101)

Professor: M. Turcotte

February 2002, duration: 75 minutes

Identification

Student name: last name: ______________________ initials: ______________________

Section: ______________________ Student number: ______________________

Signature: ____

Instructions

1. This is a closed book examination.
2. No calculators or other aids are permitted.
3. Write your answers in the space provided. Use the backs of pages if necessary.
   You may **not** hand additional pages.
4. Write comments and assumptions to get partial marks.
5. Beware, poor hand writing can affect grades.
6. Do not remove the staple holding the examination pages together.
7. This midterm test is worth 30% of the grade.

Marking scheme

Question	Maximum	Marks
1	12	
2	18	
3	20	
4	15	
5	15	
Total	**80**	

Question 1 (12 marks)

Given the following class definitions:

```
public class AAA {
    private int a = 1;
    public int get () { return a; }
}

public class BBB  extends AAA {
    private int b = 2;
    public int get () { return b; }
}

public class CCC  extends AAA {
    private int c = 3;
    public int get () { return c; }
}
```

(a) Which of the following declarations are valid?

i) `AAA a1 = new BBB();`

ii) `AAA a2 = new AAA();`

iii) `BBB a3 = new AAA();`

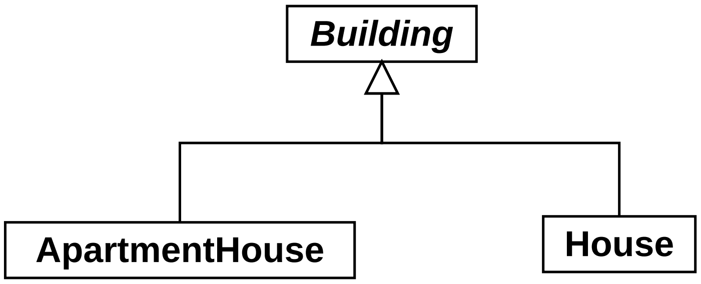
iv) `CCC a1 = new BBB();`

(b) Give the value of `x` and `y` after the execution of the following sequence of statements:

```
AAA o1 = new CCC ();
CCC o2 = (CCC) o1;
int x,y;
x = o1.get ();
y = o2.get ();
```

Question 2 (18 marks)

Given the following class hierarchy, where an `ApartmentHouse` and a `House` are specializations of a `Building`:



Knowing that:

1. all buildings have an address (a character string)
2. all buildings have a resale price (an integer)
3. the apartment houses have a number of apartments (an integer)
4. the houses have a swimming pool or not (a boolean)
5. all buildings are taxed by their city
6. the tax amount for an apartment house is equal to the resale price divided by the number of apartments.
7. the tax amount for a house is calculated as the resale price divided by 100, plus $100 for houses having a swimming pool.

(a) Write these three classes in Java including the constructors and the methods that are necessary to calculate the tax amount to be payed for a building (10 marks).

Question 2 (continued)

Question 2 (continued)

(b) Declare an array, called `city`, to hold up to 100 buildings (3 marks).

(c) Knowing there are `n` buildings in the array `city`, write a `for` loop allowing to calculate the total amount of the taxes that these buildings will have to pay (5 marks).

Question 3 (20 marks)

Given a class `Time` with the following instance method:

```
boolean before (Time t);
```

which returns `true` if this time occurs before the time `t`. Also, given a `Time` linked list whose nodes are defined as follows:

```
private class Node {

    Time value;
    Node next;

    Node (Time value, Node next) {
        this.value = value;
        this.next = next;
    }
}
```

Write an instance method that has the following declaration:

```
public boolean isIncreasing ();
```

and returns true if each `Time` element of the list occurs before the `Time` element that immediately follows it in the list. You can assume there exist an instance varialble called head that refers to the first element of the list. The empty list and the singleton list are considered to be increasing.

Question 3 (continued)

Question 4 (15 marks)

What will be printed on the screen if the method `main` of the class `Lis`, presented in the following pages, is executed?
*Read carefully **all** the methods before answering!*

```
public class Lis {

    private Object[] obj;
    private int n = 0;

    public Lis (int capInit) {
        obj= new Object[capInit];
    }

    public int dim() {
        return n;
    }

    public void ajouteAdd (Object ob) {
        if (n >= obj.length) {

            Object[] tmp= obj;

            obj= new Object[tmp.length * 3 / 2];

            for (int i=0; i<n; i++) {

                obj[i]= tmp[i];
            }
        }

        obj[n++]= ob;
    }

    public void enleveRemove() {
        if (n == 0)
            return;

        obj[--n] = null;

        if (n < obj.length/2) {

            Object[] tmp = obj;
            obj = new Object[1+tmp.length/2];

            for (int i=0; i<n; i++) {
                obj[i]= tmp[i];
            }
        }
    }
    public void afficheDisplay () {

        for (int i=0; i<obj.length; i++) {
            if (obj[i] == null)
                System.out.println ("[" + i + "] = null");
            else
                System.out.println ("[" + i + "] = " + obj[i]);
        }
    }

    public static void main (String[] arg) {
        Lis test = new Lis (4);

        test.ajouteAdd ("paul");
        test.ajouteAdd ("eve");
        test.ajouteAdd ("sam");

        System.out.println ("a)");
        test.afficheDisplay ();

        test.ajouteAdd ("tim");
        test.ajouteAdd ("amanda");

        System.out.println ("b)");
        test.afficheDisplay ();

        test.enleveRemove ();
        test.enleveRemove ();
        test.enleveRemove ();

        System.out.println ("c)");
        test.afficheDisplay ();
    }
}
```

Question 5 (15 marks)

Write a new method for the class `Lis`, described in the previous question, that removes duplicated elements from the list.

For example, if the list contains the following character strings:

```
"paul" "paul" "eve" "paul" "eve" "paul" "tim"
```

A call to the new method, called removeDuplicates, would change the list to:

```
"paul" "eve" "tim"
```

The strategy to be used is as follows:

1. A new array is created
2. For each element in the old array, check if an equivalent element already exists in the new array:
   (a) If not, this element is added at the end of the new array
   (b) If yes, this element is not added and the method proceeds with the next element
3. The new array becomes the array of the list `Lis`.

Fill in the missing elements in the method `removeDuplicates` below:

```
public void removeDuplicates () {

    Object[] tmp = obj;
    obj = new Object[tmp .length];

    int last = 0;

    for (int i=0;_______________; i++) {
        boolean found = false;

        for (int j=0; j < last && !found; j++) {
            if (tmp[i].equals(______________))
                found = true;
        }
        if (! found) {
            obj[____________] = tmp[__________];
            last++;
        }
    }
    ______________________;
}
```

(blank space)