Student ID: .....

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS – 2013

TRIMESTER 2

COMP 102

INTRODUCTION TO COMPUTER PROGRAM DESIGN

Time Allowed: THREE HOURS ******** WITH SOLUTIONS **********

Instructions: Attempt ALL Questions.

Closed book.

The exam will be marked out of 180 marks.

Silent non-programmable calculators or silent programmable calculators with their memories cleared are permitted.

Printed foreign language dictionaries are permitted.

Java Documentation will be provided with the exam script

No other material is permitted.

Answer in the appropriate boxes if possible — if you write your answer elsewhere, make it clear where your answer can be found.

There are spare pages for your working and your answers in this exam, but you may ask for additional paper if you need it.

Questions

	Marks
1. Understanding Java	[60]
2. Defining Classes	[20]
3. Files	[25]
4. Interface classes	[20]
5. 2D Arrays	[20]
6. ArrayLists of Objects	[35]

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Question 1. Understanding Java [60 marks]

(a) [8 marks] What will the following fragment of Java print out?

```
int x = 1;
int y = x;
while (y < 10){
    y = y * 2;
    UI.println(x +" : " + y);
    x = x + 1;
}
UI.println("done");
```

```
1 :  2
2 :  4
3 :  8
4 :  16
done
```

(Question 1 continued on next page)

(Question 1 continued)

(b) [9 marks] The expressionTest method below has one parameter.

```
public void expressionTest(int value){
    if ( value > 0 && value <= 5 ) {
        UI.print("first ");
    }

    if ( value < 7 || value > 9 ) {
        UI.print("second ");
    }
    else {
        UI.print("third ");
    }
    UI.println ();

}
```

What would the following calls to expressionTest print out?

expressionTest(5); $\Longrightarrow$ first second

expressionTest(6); $\Longrightarrow$ second

expressionTest(7); $\Longrightarrow$ third

(Question 1 continued on next page)

(Question 1 continued)

(c) [10 marks] Complete the following totalRent method to compute and return the total rent of a holiday house. It should have two parameters (one double and one integer) which are the weekly rent of the house and the number of weeks. If the house is rented for more than 10 weeks, it gives a 10% discount.

```
public double totalRent (double rate, int weeks){
    if (weeks > 10)
        rate = rate * 0.9;
    return (rate * weeks);

}
```

(d) [10 marks] What will be printed in the text pane if the following numbers method is called with the argument 4:

```
public void numbers (int length){
    for (int i = 0; i < length-1; i++){
        for (int j = i+1; j < length; j++){
            UI.print("("+i+","+j+") ");
        }
        UI.println(" break");
    }
    UI.println("end");
}
```

```
(0,1) (0,2) (0,3) break
(1,2) (1,3) break
(2,3) break
end
```

(Question 1 continued on next page)

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

(Question 1 continued on next page)

(Question 1 continued)

(e) [10 marks] Consider the following modifyArray method.

```
public void modifyArray(int [ ] data){
    int n=0;
    UI. println (data [3]);
    UI. println (data[n+2]);
    UI. println (data[data[n ]]);
    for( int i=0; i<data.length; i++){
        if ( data[i ] < 10){
            data[i ] = data[i ] + 10;
        }
    }
    for( int i=0; i<data.length; i++){
        UI. println (data[i ]);
    }
}
```

Suppose that the variable data is defined as follows:

```
int [ ] data = new int [ ]{4, 3, 5, 2, 16, 4, 20};
```

data:

4	3	5	2	16	4	20
0	1	2	3	4	5	6

What will be printed out by calling modifyArray(data);

```
2
5
16
14
13
15
12
16
14
20
```

(Question 1 continued on next page)

(Question 1 continued)

(f) [13 marks] The Car class below defines Car objects. Each object has three fields to store the make of the car, the year the car is made and the price. The class defines a constructor and four methods.

```
public class Car{
    // fields
    private String make;
    private int year;
    private double price;
    // constructor
    public Car(String make, int year, double price){
        this.make = make;
        this.year = year;
        this.price = price;
    }
    //methods
    public String getMake() {
        return this.make;
    }
    public int getYear(){
        return this.year;
    }
    public void putOnSale(double n) {
        this.price = this.price*(1-n);
    }
    public void printDetails (){
       UI. println (this.make + "  "+ this.year +"  " + this.price );
    }
}
```

(Question 1 continued)

What will the following testCar method print out?

```
public static void testCar(){

    Car c1 = new Car("Toyota", 2010, 10000);
    Car c2 = new Car("Ford", 2012, 20000);

    c1. printDetails ();

    c2.putOnSale(0.1);
    c2. printDetails ();

    UI. println (c1.getYear ());
    UI. println (c2.getMake());
}
```

```
Toyota 2010 10000.0
Ford 2012 18000.0
2010
Ford
```

Question 2. Defining Classes [20 marks]

For this question, you are to complete part of a program for a computer game involving a collection of buses. The buses are drawn on the screen as filled rectangles. They can turn left or right, and move to the left or to the right.

Complete the Bus class on the facing page which defines Bus objects. You need to define

- Fields: to store the Bus Number (No.), the coordinates, the direction (default "right") of the bus.
- The constructor: initialises the Bus. Parameters are the Bus Number and the x and y coordinates of the *left top corner* of the Bus. It should also draw the bus on the screen.
- The draw method: No parameters. Draw the bus as a filled rectangle with default width of 50 and height of 10;
- The erase method: No parameters. Erase the bus.
- The getNo method: No parameters. Return the Bus Number.
- the turn method: No parameters. Change the direction of the bus from left to right or vice versa. It does not draw the bus.
- The move method: Parameter is the number of pixels to move by. The bus should move in the direction that is specified in the direction field.

As an example of using the Bus class, the testBus method below creates a Bus object with a Bus Number of 4 at position (20, 100), makes it move, turn and move, prints its Bus Number and then makes it turn and move again.

```
public static void testBus(){
    Bus t = new Bus(4, 20, 100);

    t.move(200);

    t.turn();
    t.move(50);

    UI.println(t.getNo());

    t.turn();
    t.move(150);
}
```

(Question 2 continued)

```
public class Bus{
    private int no;
    private double x;
    private double y;
    private String dir = "right";

    public Bus(int n, double x, double y){
        this.no = n;
        this.x = x;
        this.y = y;
    }   this.draw();
    public void draw(){

        UI. fillRect (this.x, this.y, 50, 10);
    }
    public void erase() {
        UI.eraseRect(this.x−5, this.y−5, 60, 20);

    }
    public int getNo(){
        return this.no;

    }
    public void turn(){

        if (this.dir.equals("left"))
            this.dir = "right";
        else
            this.dir = "left";
    }


    }
    public void move(double step){

        this.erase();
        if (this.dir.equals("left"))
            this.x = this.x − step;
        else
            this.x = this.x + step;

        this.draw();
    }
```

Question 3. Files [25 marks]

Suppose the file `allcars.txt` contains data about cars at a car dealer. Each line of the file contains the model, the year, and the price of a car. For the first question, you may assume this file has only the following six lines.

```
Starlet     2001  8000
Demio       2006  14000
Accord      2004  130000
Axela       2008  17000
Accord      2008  140000
Mondeo      2009  150000
```

Assume the model name is always a single word.

(a) [10 marks] What will the following printSomething method print out?

```
public void printSomething(){

    try{
        Scanner sc = new Scanner (new File ("allcars.txt"));
        while (sc.hasNext()){
            String model = sc.next();
            int year = sc.nextInt ();
            double price = sc.nextDouble();
            if (model.equals("Accord"))
                UI. println (year);
            else
                UI. println (price );
        }
        sc.close ();
    }catch(IOException e){UI.println("fail");}
}
```

```
8000
14000
2004
17000
2008
150000
```

(Question 3 continued)

(b) [15 marks] The car dealer wants to split the list of cars into two separate files: cars under $30,000 and cars $30,000 or more.

Complete the following splitFile method that will construct two files called `cheapcars.txt` and `premiumcars.txt`, that contain all the low priced cars (under 30000) and all the high priced cars (30000 and over) from the `allcars.txt` file. The original file may contain any number of cars. You may assume that the file is correctly formatted.

Hint: Your method should open the two new files for writing, then read through `allcars.txt`, writing each line to the appropriate output file.

```
public void splitFile (){
    try{
        Scanner sc = new Scanner (new File ("allcars.txt"));
        PrintStream cheap = new PrintStream(new File("cheapcars.txt"));
        PrintStream premium = new PrintStream(new File("premiumcars.txt"));
        while (sc.hasNext()){
            String line = sc.nextLine();
            Scanner linesc = new Scanner(line);
            linesc.next();
            linesc.next();
            if (linesc.nextDouble() < 30000){
                cheap.println( line );
            }
            else {
                premium.println( line );
            }
        }
        sc.close();
        cheap.close();
        premium.close();









    }
    catch(IOException e){UI.println("Fail: " + e);}
}
```

Question 4. Interface classes [20 marks]

The university StaffAdmin program has a collection of different kinds of staff members: lecturer, postdoc, programmer, general, senior tutor and student tutor. The different types of staff members have different responsibilities and are paid differently.

However, all the classes for different types of staff members will have the following things:

- fields to store values such as the name, office number, phone number, etc.
- a constructor for creating a new staff object.
- a printInfo method that prints all the field values.
- an assignRoom method with one parameter (an int specifying the room) that assigns a room to this staff member.
- a getPay method that returns a number specifying the amount to be paid.

The StaffAdmin program uses an interface class called Staff to define the type of all the different kinds of staff members.

(a) [8 marks] Complete the following definition of the staff interface class.

```
public interface Staff {
    public void printInfo ();

    public void assignRoom(int room);

    public double getPay();


 // Note: must not have any fields
 // Note: must not have a constructor
 // Note: must not have any method bodies




}
```

(Question 4 continued on next page)

(Question 4 continued)

(b) [12 marks] The following Programmer class describes Programmer objects, and provides several methods for Programmers.

However, Programmer is not correctly written to make every Programmer object also be a Staff. Complete/alter the definition of Programmer below so that a Programmer is also a Staff.

```
public class Programmer { X   implements Staff {

    private String name;
    private int room;
    private double payment;
    private String jobs;

    public Programmer(String n, double p){
        this.name = n;
        this.payment = p;
    }
    public void assignJob(String jobDescription){
        this.jobs = jobDescription;
    }
    public void assignRoom(int n){
        this.room = n;
    }



    // Note: all the methods in Staff must be defined.
    public double getPay(){
        return this.payment;
    }
    public void print(){
        UI.println(this.name);
        UI.println(this.room);
        UI.println(this.jobs);
        UI.println(this.payment);
    }









}
```

Question 5. 2D Arrays [20 marks]

A matrix is a 2D rectangular array of numbers. There are many applications of matrices and many different operations on them. For this question, you are to implement two methods that operate on matrices.

(a) [10 marks] Complete the following saveMatrix method that saves the values in a matrix to a file, lined up in rows and columns. For example, if the matrix has two rows and 3 columns, it might save it as follows in a file:

```
  6.23   15.41  -13.45
-27.18   -3.87   85.00
```

The method should save each value with two decimal places, ensuring that there are at least two spaces between numbers. You may assume that every number in the matrix can be saved with at most 6 characters.

Assume that the first index represents the row, and the second index represents the column.

The method has two parameters: the matrix and the file name.

```
public void saveMatrix(double [ ][ ] matrix, String fileName){

   try{
      PrintStream out = new PrintStream(new File(fileName));
      for (int r =0; r<matrix.length; r++){
        for (int c =0; c<matrix[r].length; c++){
            //UI.printf("  %6.2f", matrix[r][c]);
            out.printf(" %6.2f", matrix[r][c]);
        }
        //UI.println();
        out.println();
      }
    out.close();









   }
   catch(Exception e){UI.println("file error");}
}
```

(Question 5 continued on next page)

(Question 5 continued)

(b) [10 marks] Subtracting a matrix from another matrix of the same size involves subtracting each element of the matrix from the corresponding element in the other matrix.

For example, if A is the matrix

$$\begin{pmatrix} 10 & 10 & 10 \\ 4 & 5 & 6 \end{pmatrix}$$

then subtracting the matrix

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 4 & 4 \end{pmatrix}$$

will change A to

$$\begin{pmatrix} 9 & 8 & 7 \\ 0 & 1 & 2 \end{pmatrix}$$

Complete the following subtractMatrix method that is passed two matrices, M and S, and will subtract S from M. The method modifies the contents of matrix M; it does not need to return anything. The method should first check that M and S are the same size, and only modify M if they are the same size.

```
public void subtractMatrix(double[][] M, double[][] S){
    if (M.length == S.length && M[0].length == S[0].length){
        for (int r =0; r<M.length; r++){
            for (int c =0; c<M[r].length; c++){
                M[r][c] = M[r][c] - S[r][c];
            }
        }
    }
}
```

Question 6. ArrayLists of Objects [35 marks]

This question concerns the CarDatabase program to keep track of the cars for a car dealer. The CarDatabase class (first few lines are shown on the facing page) stores details of each car in an ArrayList of Car objects.

The Car class, shown below, defines Car objects that store information about individual cars, including their make (e.g Toyota, Ford), year and price. This class is the same as the Car class used in Question 1.

```
public class Car{
    // fields
    private String make;
    private int year;
    private double price;
    // constructor
    public Car(String make, int year, double price){
        this.make = make;
        this.year = year;
        this.price = price;
    }
    //methods
    public String getMake() {
        return this.make;
    }
    public int getYear(){
        return this.year;
    }
    public void putOnSale(double n) {
        this.price = this.price*(1-n);
    }
    public void printDetails (){
       UI.println (this.make + "("+ this.year +") : $" + this.price );
    }
}
```

(Question 6 continued)

First few lines of the CarDatabase class:

```
private ArrayList <Car> cars = new ArrayList<Car>();

public void listCars (){
    for (Car a : this.cars){
        a.printDetails ();
    }
}
```

(a) [8 marks] Complete the following addCar method of the CarDatabase class. It has a parameter - a car which should be added to the end of the ArrayList of cars. It should also print a "car added" message and the number of cars on the list.

```
public void addCar(Car c){
    this.cars.add(c);
    UI.println("Car added");
    UI.println("Number of cars: " + this.cars.size());

}
```

(b) [10 marks] Complete the following oldCarForSale method with two parameters – a year and a discount. The method should find all the cars that are made before the specified year, put them on sale (call the right method in the Car class to do this), and print their details.

```
public void oldCarForSale (int year, double d){
    for (Car c : this.cars) {
        if (c.getYear() < year){
            c.putOnSale(d);
            c.printDetails ();
        }
    }



}
```

(Question 6 continued)

(c) [17 marks] Complete the following clusterCars method to put the cars with the same `make` together. For example, if the original list is this:

```
Toyota 1991 3000.0
Toyota 1999 10000.0
Ford 1990 2000.0
Ford 1991 1000.0
Toyota 1995 5000.0
Ford 1994 2500.0
Honda 1993 3000.0
Toyota 2013 30000.0
Toyota 1999 10000.0
Honda 1993 2000.0
```

then the new list should be:

```
Toyota 1991 3000.0
Toyota 1999 10000.0
Toyota 1995 5000.0
Toyota 2013 30000.0
Toyota 1999 10000.0
Ford 1990 2000.0
Ford 1991 1000.0
Ford 1994 2500.0
Honda 1993 3000.0
Honda 1993 2000.0
```

There are many different ways of doing this. One way is to create a new ArrayList to save the cars in this order, and then at the last step to assign this new ArrayList back to the original cars data field.

(Question 6 continued)

```
public void clusterCars(){

        ArrayList<Car> newList = new ArrayList<Car>();
        while(!this.cars.isEmpty()){
          String m = this.cars.get(0).getMake();
          newList.add(this.cars.get(0));
          this.cars.remove(0);
          for (int i = 0; i< cars.size(); i++){
            if (cars.get(i).getMake().equals(m)){
                newList.add(this.cars.get(i));
                this.cars.remove(i);
                i = i-1;
            }
          }
        }
        this.cars = newList;









}
```

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

****