TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS – 2015

TRIMESTER 1

COMP 102
INTRODUCTION TO COMPUTER PROGRAM DESIGN

Time Allowed: TWO HOURS ******** WITH SOLUTIONS **********

CLOSED BOOK

Permitted materials: Silent non-programmable calculators or silent programmable calculators with their memories cleared are permitted in this examination.

Printed foreign language to English dictionaries are permitted.

No other material is permitted.

Instructions: Attempt ALL Questions.

The exam will be marked out of 120 marks.

Brief Java Documentation will be provided with the exam script

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. If, While and Array	[20]
2. Defining two classes	[20]
3. Files	[22]
4. ArrayLists of Objects	[35]
5. 2D Arrays	[23]

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. If, While and Array [20 marks]

(a) [4 marks] What will the following printIt method print out?

```
public void printIt (){
    int x = 20;
    while (x < 40){
        x = x + 5;
        UI.println ("x:" + x);
    }
    UI.println ("finally "+ x);
}
```

x:

```
x:  25
x:  30
x:  35
x:  40
finally 40
```

(Question 1 continued on next page)

(Question 1 continued)

(b) [4 marks] The testIt method below has one parameter.

```
public void testIt ( int x){

    if ( x > 8 ) {
        if ( (x > 10) && (x < 15) ) {
            UI.print ("Good");
        }
        else {
            UI.print ("Better ");
        }
    }
    else if ( (x > 20) || (x < 6) ){
        UI.print ("Best ");
    }
    else {
        UI.print ("Bad");
    }
}
```

What would the following calls to testIt print out?

testIt(4); ⟹ Best

testIt(17); ⟹ Better

testIt(9); ⟹ Better

(Question 1 continued on next page)

(Question 1 continued)

(c) [5 marks] Consider the following printArray method which is passed an array of integers.

```
public void printArray( int[ ] nums){

    for ( int i = 1; i <nums.length−1; i++){
        UI. print (nums[i]);
    }
    UI. println ();

    nums[1]=nums[3]+ nums[1];
    UI. println ("1st: " + nums[1]);
    UI. println ("2nd: " + nums[3]);
    UI. println ("3rd: " + nums[nums[3]]);

    for ( int i = nums.length−1; i >= 0; i=i−2){
        UI. print (nums[i]);
    }
    UI. println ();
}
```

What will the PrintArray method print out if it is passed the following array?

myNums:

3	4	2	5	9	1	0
0	1	2	3	4	5	6

```
42591
1st:  9
2nd:  5
3rd:  1
0923
```

(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)

(d) [7 marks] Complete the following printNames method which is passed an array of Strings. This method should print to the UI text pane all the Strings stored in the array. If a String is too long (more than 20 characters), then the program should only print the first 20 characters of the String.

Hint: Consider the substring method in the String class.

```
public void printNames(String [ ] names){
        for (int i =0; i< names.length; i++) {
            if (names[i].length()<=20)
                UI. println (names[i]);
            else
                UI. println (names[i].substring (0,20));
        }
  //OR
        for (String name : names) {
            if (name.length()<=20)
                UI. println (name);
            else
                UI. println (name.substring(0,20));
        }



}
```

Question 2. Defining two Classes [20 marks]

This question asks you to complete two classes Shareholder and Company to store information about shareholders of companies.

(a) [10 marks] Complete the Shareholder class on the opposite page.

A Shareholder object should contain two fields:

- name: the name of the shareholder.
- shares: the number of shares they have.

Shareholder should have a constructor that takes a name and a number and stores them in the two fields.

Shareholder should have three methods:

- addShares, which is passed the number of new shares and adds them to the current number of shares.
- getShares, which returns the current number of shares.
- toString, which returns a string containing the values of the two fields in the form "John Smith (3400)".

(Question 2 continued on next page)

(Question 2 continued)

```
public class Shareholder {
    // fields
    private String name;
    private int shares;



    // constructor
    public Shareholder(String n, int s){
        this.name = n;
        this.shares = s;


     }
     //methods

    public void addShares(int num) {
        this.shares = this.shares + num;
    }

    public int getShares() {
        return this.shares;
    }

    public String toString () {
        return this.name +"(" + this.shares +")";
    }









}
```

(Question 2 continued on next page)

(Question 2 continued)

(b) [10 marks] Complete the Company class on the opposite page.

A Company object should contain two fields:

- name: the name of the company
- shareholders: a list of Shareholders.

The Company constructor should have one parameter (the company name), and should initialise both fields.

Company should have two methods:

- addShareholder, which is passed a name (of the shareholder) and a number (number of shares). It should create a Shareholder object and add it to the list of shareholders.
- primaryOwner, which finds the Shareholder who holds the biggest number of shares, prints his/her details including name and the number of shares.

(Question 2 continued on next page)

(Question 2 continued)

```
public class Company {
    private String name;
    private ArrayList<Shareholder> shareholders;


    public Company(String n){
        this.name = n;
        this.shareholders = new ArrayList<Shareholder>();

    }
    public void addShareholder(String n, double s) {
        this.shareholders.add(new Shareholder(n, s));




    }
    public void primaryOwner() {
        int d = 0;
        Shareholder sh = null;
        for(Shareholder s: this.shareholders) {
            if ( s.getShares() > d ){
                d = s.getShares();
                sh = s;
            }
        }
        UI. println (sh.toString ());        // OR    UI.println(sh);
```

```
    }
}
```

Question 3. Files [22 marks]

Suppose a web site has information for rental accommodation that can be downloaded as a text file. Each line of the file contains the location, size, rent and comment for one accommodation.

For example, the downloaded file might contain:

```
Kelburn 2 br $400 pw sunny flat
Te Aro 1 br $450 pweek house
Karori 3 bdrms home $600 pw short term only
Lyall Bay 2 br flat 300 pw flat mate wanted
Northland 1 br flat $280 perweek furnished
```

(a) [8 marks] What will the following printData method print out if called with the name of the downloaded file above?

```
public void printData(String fileName){
    try{
        Scanner sc = new Scanner(new File(fileName));

        UI.println(sc.next());
        UI.println(sc.nextInt());
        sc.next();
        UI.println(sc.next());
        String info = sc.nextLine();

        UI.println("--------------------");

        while ( sc.hasNext() ){
            if (sc.hasNextInt()){
                int num = sc.nextInt();
                UI.println("num: " + num);
            }
            else {
                sc.next();
            }
        }

        sc.close();
    } catch(IOException e){UI.println("file reading failed"+e);}
}
```

(Question 3 continued on next page)

(Question 3 continued)

Write your answer here:

```
kelburn
2
$400

------------------

num:  1
num:  3
num:  2
num:  300
num:  1
```

(Question 3 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 3 continued)

(b) [14 marks] Complete the following findLarge method, whose first parameter is the name of a file, and second parameter is the minimum size of the accommodation. The findLarge method should read the file and print out the lines describing an accommodation that is at least as big as the given size.

E.g., if **data.txt** is the name of the file on the previous page, findLarge("data.txt", 2) should print out

```
Kelburn 2 br $400 pw sunny flat
Karori 3 bdrms home $600 pw
Lyall Bay 2 br flat 300 pw flat mate wanted
```

You may assume that

- the location is either one or two words, but never more than two words;
- the size (the number of bedrooms) is always an integer

```
public void findLarge(String filename, int min){
    try{
        Scanner sc = new Scanner(new File(filename));
        while (sc.hasNext()){
            String loc = sc.next();
            if (!sc.hasNextInt()){
                loc = loc + "  " + sc.next();
            }
            int size = sc.nextInt();
            String info = sc.nextLine();
            if (size >= min){
                UI.println(loc + "  " + size + info);
            }
        }
        sc.close();









    } catch(IOException e){UI.println("file reading failed"+e);}
}
```

Question 4. ArrayLists of Objects [35 marks]

This question concerns a program for managing a database of jokes posted at a web site. The program consists of a Joke class for representing individual jokes and a JokeApp class which stores a list of jokes in an ArrayList. Users can vote for a joke by giving "thumb-Ups" and "thumb-Downs" and these votes are stored in the Joke object.

The Joke class below defines Joke objects, which store the text of the joke and the number of Thumb-Up votes and Thumb-Down votes.

```
public class Joke {
    private String jokeText;
    private int thumbUps = 0;
    private int thumbDowns = 0;

    public Joke(String txt) {
        this.jokeText = txt;
    }

    public String getText(){
        return this.jokeText;
    }
    public String toString(){
        return this.jokeText + " (ups: " + this.thumbUps + ") (downs: " + this.thumbDowns + ")";
    }

    public int getUps() {
        return this.thumbUps;
    }
    public int getDowns() {
        return this.thumbDowns;
    }
    public double rating() {
        return 1.0*(this.thumbUps - this.thumbDowns)/(this.thumbUps + this.thumbDowns);
    }

    public void voteUp() {
        this.thumbUps ++;
    }
    public void voteDown() {
        this.thumbDowns++;
    }
}
```

(Question 4 continued on next page)

(Question 4 continued)

The JokeApp class contains a field to store the list of Joke objects, and several methods that search or modify the list of jokes.

```
public class JokeApp {
    private ArrayList<Joke> jokes = new ArrayList<Joke>();

    /** Create a new Joke and add to the front of the list */
    public void addNewJoke(String text){
        this.jokes.add(0, new Joke(text));
    }
```

Assume that jokes will never contain any null values.

(a) [6 marks] Complete the following countPopular method which will return the number of jokes in the list that have more than 20 votes (counting both thumb-Up and thumb-Down votes).

```
public int countPopular(){
    int count = 0;
    for (Joke joke : this.jokes) {
        if (joke.getDowns() + joke.getUps() > 20){
            count++;
        }
    }
    return count;




}
```

(Question 4 continued on next page)

(Question 4 continued)

(b) [6 marks] Complete the following allPositive method which returns true if every joke in the list has more thumb-Up votes than thumb-Down votes, and false otherwise. It should return true if the list contains no jokes.

```
public boolean allPositive (){
    for (Joke joke : jokes) {
        if (joke.getUps() <= joke.getDowns()){
            return false;
        }
    }
    return true;




}
```

(Question 4 continued on next page)

(Question 4 continued)

(c) [11 marks] Complete the following removeBad method which will remove from the list every joke that has more thumb-Down votes than thumb-Up votes.

Hint: Remember that removing a value from a list shifts all the later values down one place!

```
public void removeBad(){
    for ( int i=jokes.size()-1; i>=0; i--) {
        Joke j = this.jokes.get(i);
        if (j.getDowns() > j.getUps()){
            this.jokes.remove(i);
        }
    }



}
```

(Question 4 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.

Two alternative answers:

```
        int bestIndex = 0;
        for (int i = 0; i< this.jokes.size(); i++){
            if (this.jokes.get(i).rating() > this.jokes.get(bestIndex).rating()){
                bestIndex = i;
            }
        }
        this.jokes.add(0, this.jokes.remove(bestIndex));
        int worstIndex = 0;
        for (int i = 0; i< this.jokes.size(); i++){
            if (this.jokes.get(i).rating() < this.jokes.get(worstIndex).rating()){
                worstIndex = i;
            }
        }
        this.jokes.add(this.jokes.remove(worstIndex));
OR ----------------------------------------------------------------
        // more efficient, but more complicated
        int bestIndex = -1;
        int worstIndex = -1;
        double bestRating = -2;   // anything less than lowest possible rating = -1.0
        double worstRating = 2;   // anything more than highest possible rating = 1.0
        for (int i = 1; i < this.jokes.size(); i++){
            double rating = this.jokes.get(i).rating();
            if (rating > bestRating) {
                bestRating = rating;
                bestIndex = i;
            }
            if (rating < worstRating) {
                worstRating = rating;
                worstIndex = i;
            }
        }
        jokes.add(jokes.remove(worstIndex));
        if (bestIndex < worstIndex){
            jokes.add(0,jokes.remove(bestIndex));
        }
        else {
            jokes.add(0,jokes.remove(bestIndex-1));
    }
```

(Question 4 continued)

(d) [12 marks] Complete the following bestAndWorst method so that it moves the highest rated joke to the front of the list and moves the lowest rated joke to the end of the list. The order of the other jokes should not be changed.

The rating of a joke is given by

$$\text{rating} = \frac{(\text{thumbUps} - \text{thumbDowns})}{(\text{thumbUps} + \text{thumbDowns})}$$

Hint: There is a rating method in the Joke class.

```
public void bestAndWorst(){
        Joke best = this.jokes.get(0);
        Joke worst = best;
        for (Joke joke : this.jokes){
            if (joke.rating() > best.rating()){
                best = joke;
            }
            if (joke.rating() < worst.rating()){
                worst = joke;
            }
        }
        jokes.remove(best);              // assumes that the jokes are all unique
        jokes.add(0,best);
        jokes.remove(worst);
        jokes.add(worst);









}
```

Question 5. 2D arrays [23 marks]

This question concerns a seat booking program for a theatre. The theatre has 20 rows and each row has 40 seats. The premium seats are the middle 20 seats (seat numbers 10-29) in the front rows (rows 0 - 5).

The program uses a 2D array of Strings to record the seat bookings. Each array element contains the name of the person who has booked the seat; if it contains null, then it has not been booked.

The program has several constants for the display, and uses a field named bookings to store the seat bookings:

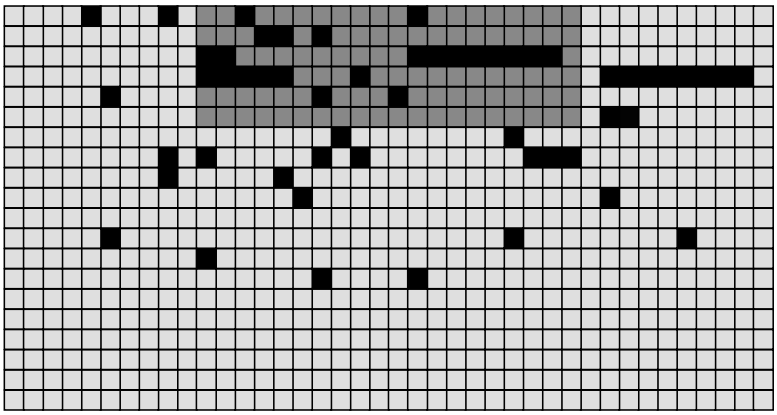
```
public static final int LEFT = 20; // left side of the seat display
public static final int TOP = 30;  // top of the seat display
public static final int SIZE = 10; // size of each seat

private String[ ][ ] bookings = new String[20][40];
```

(a) [8 marks] Complete the following displaySeats method to display the seats and bookings as a grid of squares in the graphics pane.

- Seats that are booked should be shown as squares filled with black.
- Premium seats that are available should be shown as squares filled with red.
- Other seats that are available should be shown as squares filled with yellow.
- The 0'th row should be displayed at the top.

An example is shown below.



(Question 5 continued on next page)

(Question 5 continued)

```
public void displaySeats(){
    for( int row = 0; row < this.bookings.length; row++){
        int y = TOP + row*SIZE;
        for( int seat = 0; seat < this.bookings[row].length; seat++){
            int x = LEFT + seat*SIZE;
            if (this.bookings[row][seat]!=null) {    // set the right colour
                UI.setColor(Color.black);
            }
            else if (row <= 5 && seat >=10 && seat <= 29) {
                UI.setColor(Color.red);
            }
            else {
                UI.setColor(Color.yellow);
            }
            UI. fillRect (x, y, SIZE, SIZE);        // fill with colour

            UI.setColor(Color.black);
            UI.drawRect(x, y, SIZE, SIZE);        // draw outline
        }
    }
OR
    UI.setColor(Color.yellow);
    UI. fillRect (LEFT, TOP, SIZE*40, SIZE*20);        // make it all yellow
    UI.setColor(Color.red);
    UI. fillRect (LEFT+10*SIZE, TOP, SIZE*20, SIZE*6); // make premium red
    UI.setColor(Color.black);
    for( int row = 0; row < ROWS; row++){              // empty/filled squares
        for( int seat = 0; seat < this.bookings[row].length; seat++){
            if (this.bookings[row][seat]==null) {
                UI.drawRect(LEFT + seat*SIZE, TOP + row*SIZE, SIZE, SIZE);
            }
            else {
                UI. fillRect (LEFT + seat*SIZE, TOP + row*SIZE, SIZE, SIZE);
            }
        }
    }
}
```

(Question 5 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 5 continued)

(b) [15 marks] Complete the following blockBooking method which is given a desired number of seats and a name. It then searches for a block of that many free seats in a single row. If it finds such a block, it books them under that name, and returns true to signal success. If it can't find such a block, it returns false.

To get the last 2 marks, your code should find a block of premium seats if possible.

```
public boolean blockBooking(int count, String name){
    if (count > this.bookings[0].length || count < 1) {
        return false;
    }
    for (int row=0; row<this.bookings.length; row++){
        for (int start=0; start<=this.bookings[row].length-count; start++){
            boolean found = true;
            for (int i=0; i<count; i++){
                if (this.bookings[row][start+i]!=null){
                    found = false;
                }
            }
            if (found){
                for (int i=0; i<count; i++){
                    this.bookings[row][start+i] = name;
                }
                return true;
            }
        }
    }
    return false;
OR
    for (int row=0; row<this.bookings.length; row++){
        int empty = 0;
        for (int seat=0; seat<this.bookings[0].length; seat++){
            if (this.bookings[row][seat]!=null){ empty = 0; }
            else {
                empty++;
                if (empty==count){
                    for (int i=0 ; i<empty; i++){
                        this.bookings[row][seat-i] = name;
                    }
                    return true;
    }}}}
    return false;

}
```

* * * * * * * * * * * * * * *