TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VUW VICTORIA UNIVERSITY OF WELLINGTON

EXAMINATIONS – 2024

TRIMESTER 1

COMP 102
INTRODUCTION TO COMPUTER PROGRAM DESIGN

Time Allowed: TWO HOURS

CLOSED BOOK ******** **WITH SOLUTIONS** **********

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

Printed foreign language–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:

1. Understanding Java [27]
2. Design a class [20]
3. Files [23]
4. ArrayLists of Objects [30]
5. Arrays and 2D Arrays [20]

May 7, 2024

Student ID: .....

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 **[27 marks]**

(a) **[4 marks] Calling methods.**

What will the following printStuff method print out?

```
public void printStuff (){
    UI.println("printing stuff");
    int x = 10;
    int y = 15;
    int z = y * 2;
    UI.println("z=" + z);
    z = this.myMethod(x, y);
    UI.println("x=" + x + "  y="+ y + "  z=" + z);
}

public int myMethod(int m, int n){
    m = n - m;
    return m;
}
```

```
printing stuff
z=30
x=10 y=15 z=5
```

(Question 1 continued on next page)

(Question 1 continued)

(b) **[7 marks]** Conditionals

Write a method shippingFee(...) that calculates and returns the shipping cost for a delivery address. The method should have a parameter to specify the address, and should return the cost as follows:

- Local: $10.00
- Nation-wide: $20.00
- Rural: $30.00
- International: $50.00

The address is a string and it always contains "Wellington" for local, "New Zealand" or "NZ" for nation-wide, and "RD" for rural delivery. You may assume any other address strings are international.

```
public double shippingFee(String address) {


    if (address.contains("Wellington"))
        return 10.0;
    else if (address.contains("RD"))
        return 30.0;
    else if (address.contains("New Zealand") || address.contains("NZ"))
        return 20.0;
    else
        return 50.0;
}
```

Marking scheme:
2: For each correct **return** (max 6)
1: All states **return** something
If clearly wrong:
1: Per evidence of learning (max 3)
}

(Question 1 continued on next page)

(Question 1 continued)

(c) **[7 marks] Arrays.**

What will the following arrayTest method print out?

```
public void arrayTest(){
    int [ ] nums = new int [ ]{5, 2, 4, 3, 1, 4, 9, 5};

    int n = 1;
    UI.println(nums.length);
    UI.println(nums[4]);
    UI.println(nums[n+1]);
    UI.println("-----------");

    for(int i = 2; i < 5; i++){
        nums[i] = nums[i] + nums[i-1];
        UI.println(nums[i]);
    }
}
```

```
8
1
4

-------------
6
9
10
```

(Question 1 continued on next page)

(Question 1 continued)

(d) **[9 marks] ArrayLists.**

Complete the following countStartsWithA() method. It gets some words from the user and stores them in words. It will count the number of words that start with the letter A. Finally, it prints the result.

You method should not be case sensitive.

For example, if the user enters the following words:
"An"
"accountant"
"went"
"2"
"prison"

It should print "2 words started with an A".

```
    public void countStartsWithA() {
        ArrayList<String> words = UI.askStrings("Enter words here: ");


        int num = 0;

        for(String s : words) {
            if(s.startsWith("a") || s.startsWith("A")) { //or use charAt(0)
             num++;
            }
        }

        UI.println(num + " words started with an A");



Marking: (min 5)
1: Variable to count initialised
2: Loop that loops all words
3: Conditional for both a and A
1: Increment count variable
2: Print result
−0.5: minor errors (e.g. only checking A (not a), miss first/last word in loop, only print num)

Marking: (max 4)
1: Evidence of learning


    }
```

Question 2. Design a class **[20 marks]**

Complete the following DrawBoxGUI class that allows the user to draw a single filled square on the graphics pane and then press a button to make it shrink.

- The fields should store the position and size of the square. Note, a square means the width and height are equal.
  - The code does not need to set the color of the square.
- Declare the fields so they cannot be accessed directly outside the class.
- The constructor and setupGUI() method have been written for you.
- The draw() method draws the square. It should be called every time the square has changed position or size.
- The doMouse method should draw a square at the position the user clicked, so that the middle of the square is where the user clicked (i.e. not top/left). The default starting size is 50.0.
- The shrink() method should shrink the square by 10, and redraw it. The square must never be drawn with a size of zero or lower, instead set the square's size to the default size (i.e. 50.0).
- Remember to clear the graphics pane before redrawing.

```
public class DrawBoxGUI {
    private double centerX;
    private double centerY;
    private double size;

    public DrawBoxGUI() { this.setupGUI(); }

    public void setupGUI(){
        UI.setMouseListener(this::doMouse);
        UI.addButton("Shrink", this::shrink);
    }

    public void draw() {
        //UI.ckearGraphics() can also be called here instead.
        UI.fillRect(this.centerX-this.size/2,this.centerY-this.size/2,
                    this.size, this.size);
    }

    public void doMouse(String action, double x, double y) {

        if (action.equals("clicked")){ // or "released"
            UI.clearGraphics();

            this.size = 50;
            this.centerX = x;
            this.centerY = y;
            this.draw();
        }


    }

    public void shrink() {

        UI.clearGraphics();
        this.size = this.size - 10;

        if(this.size <= 0) {
            this.size = 50;
        }
        this.draw();
    }
}
```

Marking:
6: Appropriate Fields
1: Private fields (all)
2: draw is correct
2: clear graphics correctly
2: doMouse check clicked/released
3: doMouse set pos/size correctly
1: doMouse calls draw
2: shrink change size correctly
1: shrink calls draw
−0.5: minor mistakes (including allow box size <= 0)
Marking when clearly wrong (Max 7)
1: Evidence of learning

Question 3. Files **[23 marks]**

A data file football.txt is shown below. Each line contains a team name and three integer numbers. The first integer is the number of superbowls the team has played in. The second integer is the number of superbowls the team has won, and the last value is the number of touchdowns the team has scored in the superbowls they have played in.

```
Steelers 8 6 24
Patriots 11 6 30
49ers 8 5 31
Cowboys 8 5 26
```

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

```
public void printFromFile(){
    try {
        List<String> lines = Files.readAllLines(Path.of("football.txt"));
        for(String str : lines) {
            Scanner scan = new Scanner(str);
            String s = scan.next();
            scan.next();
            scan.next();
            UI.println(s + " " + scan.next());
            scan.close();
        }
    } catch(IOException e) {UI.println("file error");}
}
```

```
Steelers 24
Patriots 30
49ers 31
Cowboys 26
```

(Question 3 continued on next page)

(Question 3 continued)

(b) **[13 marks]** Complete the following highestRate() method that will read football.txt and finds the team name with the highest rate of touchdowns per superbowl played, i.e. the last integer divided by the first integer in each line.

The result should be printed using either UI.println() or UI.printf, where the team name is printed together with the rate of touchdowns per superbowl.

For example, running highestRate() using the provided football.txt file must result in the following output:
"49ers 3.875"

```
public void highestRate(){
    try {
        List<String> lines = Files.readAllLines(Path.of("football.txt"));
    double max = Double.MIN_VALUE;
    String team = "";

    for (String line : lines) {
        Scanner sc = new Scanner(line);

        String currentTeam = sc.next();
        double first = sc.nextDouble();
        sc.next();
        double last = sc.nextDouble();

        if(last / first > max) {
            team = currentTeam;
            max = last/ first ;
        }
        sc.close();
    }

    UI.println(team + "  " + max);

Marking:
1: set up max/team
1: loop all lines
4: read line correctly (find current team, first, last, and skip second value) and close scanner
1: find last value
1: avoid integer division issue
2: check if rate is higher than already known
2: change max/team correctly
1: print result correctly
Marking when clearly wrong (max 6)
1: Evidence of learning
    } catch(IOException e) {UI.println("file error");}
}
```

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 4. ArrayLists of Objects **[30 marks]**

A program has two classes: a Song class and a Radio class.

The Song class below defines Song objects.

```
class Song {
    private String performer;
    private String name;
    private int length;   //in seconds
    private String style;


    public Song(String p, String n, int l, String s) {
        this.performer = p;
        this.name = n;
        this.length = l;
        this.style = s;
    }

    public String getName() {
        return this.name;
    }

    public String getPerformer(){
        return this.performer;
    }

    public int getLength(){
        return this.length;
    }

    public String getStyle(){
        return this.style;
    }
}
```

The Radio class declares a songs field to store the list of song objects:

```
private List<Song> songs = new ArrayList<Song>();
```

and some constants that must be used for the different styles of songs.

```
private static final String ROCK = "ROCK";
private static final String CLASSICAL = "CLASSICAL";
private static final String POP = "POP";
private static final String JAZZ = "JAZZ";
```

(Question 4 continued on next page)

(Question 4 continued)

(a) **[5 marks]** What will the following method of the Radio class print when it is executed?

```
public void testing(){
    Song s1 = new Song("ABBA", "Waterloo", 164, POP);
    Song s2 = new Song("U2", "I Will Follow", 220, ROCK);
    Song s3 = new Song("Take 6", "Milky-White Way", 287, JAZZ);

    this.songs.clear();
    this.songs.add(s1);
    this.songs.add(s2);
    this.songs.add(0, s3);
    this.songs.add(new Song("ABBA", "SOS", 202, POP));

    for(Song s: songs) {
        UI.println(s.getPerformer());
    }

    UI.println(songs.get(0).getLength());
}
```

```
Take 6
ABBA
U2
ABBA
287
```

(Question 4 continued on next page)

(Question 4 continued)

(b) **[10 marks]** Complete the following returnSongsBy(...) method that will find all songs by a particular performer, save them in a new ArrayList, and return the ArrayList.

```
public List<Song> returnSongsBy(String performer) {
        List<Song> result = new ArrayList<Song>();

        for(Song s: songs) {
            if(s.getPerformer().equals(performer)) {
                result.add(s);
            }
        }

        return result;

Marking:
1: Set up an ArrayList
2: Loop all songs
4: Check the performer equals the song's performer
-1: for each mistake (e.g. ==)
2: Add song correctly to ArrayList
1: Return the ArrayList









}
```

(Question 4 continued on next page)

(Question 4 continued)

(c) **[15 marks]** The following findSongs(...) method has three parameters which are the minimum length, maximum length, and style of songs that needs to be found.

Complete the findSongs(...) method so that it prints out all songs that have a length between the minimum and maximum length (both included) with the provided style.

For example, using findSongs after running testing() in (a), the following calls would print:
findSongs(100, 200, ROCK)
No songs found

findSongs(100,300,POP)
ABBA Waterloo
ABBA SOS

findSongs(100,200,POP)
ABBA Waterloo

```
public void findSongs( int minLength, int maxLength, String style ){
    boolean found = false;
    for(Song s: songs) {
        if (s.getLength() >= minLength && s.getLength() <= maxLength &&
            s.getStyle().equals( style )) {
            UI.println(s.getPerformer() + "  " + s.getName());
            found = true;
        }
    }
    if (!found) {
        UI.println("No songs found");
    }
}
```

Use whichever marking scheme is higher

Marking:
1: Using a found variable
2: Loop all songs
7: Check correct songs length and style fits
−1: per mistake (< instead of <=, || instead of &&, or s.getStyle()==style)
3: Print correctly if songs fits criteria
2: Print "No songs found" only when no song was found

Marking when clearly wrong (max 7)
1: Evidence of learning

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. Arrays and 2D Arrays **[20 marks]**

A motel currently has 10 rooms. They are all in one row in one building. To remember the guest's first names, they have a board on the wall similar to the one below:

Name		Thor	Freya		Tane	Ngaio		Helen	Bill	
Room	1	2	3	4	5	6	7	8	9	10

Currently, rooms 1, 4, 7, and 10 are empty.

(a) **[6 marks]** A naive programmer has provided the below code to create a computer application to substitute the board on the wall. (The line numbers are provided for convenience.)

```
[01] public class Motel {
[02]     private static final int MAX_ROOM = 10;
[03]     private String [] names = new String[MAX_ROOM];
[04]
[05]     public void assignGuest(String name, int room){
[06]         this.names[room] = name;
[07]     }
[08]
[09]     public boolean isFree(int room){
[10]         return (this.names[room]==null);
[11]     }
[12]
[13]     public String getName(int room){
[14]         return (this.names[room]);
[15]     }
[16]
[17]     public boolean checkout(String name, int room){
[18]         if (this.names[room].equals(name)){
[19]             this.names[room] = null;
[20]             return true;
[21]         }
[22]         return false;
[23]     }
[24] }
```

The other classes that use the Motel class have been tested and will always call the Motel class' methods using a room number between 1 and MAX_ROOM and valid names (i.e. they do not call any methods with null as an argument).

Still there are bugs in Motel that will make the program crash when it runs. Changing two lines will remove the bugs. Please write the line numbers below:

(Question 5 continued on next page)

(Question 5 continued)

The motel is expanding and are building several rows of rooms, so that a physical name board would look similar to this:

Row 1	Name		Thor	Freya	Mo	Tane	Ngaio		Helen	Bill	
	Room	1	2	3	4	5	6	7	8	9	10
Row 2	Name	Bob	Joe			Jane	Eva				Pia
	Room	1	2	3	4	5	6	7	8	9	10
Row 3	Name		Moana		Monique			Ghassem		Karsten	
	Room	1	2	3	4	5	6	7	8	9	10

Rows are numbered 1 to MAX_ROW.

Instead of a physical board they want an application to do this.

(Question 5 continued on next page)

(Question 5 continued)

(b) **[8 marks]** Complete MotelManyRows so that each method has the same functionalities as in (a). Make sure that the bugs in (a) are fixed. The names array must be changed to a 2d array.

```
public class MotelManyRows {
    private static final int MAX_ROOM = 10;
    private static final int MAX_ROW = 3;
    //Declare names as 2d array

    private String [][] names = new String[MAX_ROW+1][MAX_ROOM+1];

    public void assignGuest(String name, int row, int room){
        this.names[row][room] = name;

    }

    public boolean isFree(int row, int room){
        return (this.names[row][room]==null);

    }

    public String getName(int row, int room){

        return (this.names[row][room]);

    }

    public boolean checkout(String name, int row, int room){

        if (this.names[row][room] != null && this.names[row][room].equals(name)){ //logic like in (a)
            this.names[row][room] = null;
            return true;
        }
        return false;


    }
}
```

Marking:
2: new 2d Array
1: assignGuest
1: isFree
1: getName
3: checkout
  1: check names
  1: assign null
  1: return correctly
−0.5: minor mistake
−2: Crash errors still present
Marking when clearly wrong: (Max 4)
1: Evidence of learning

(Question 5 continued on next page)

(Question 5 continued)

(c) **[6 marks]** Complete the following countEmptyRooms() method that returns the number of empty rooms in the motel (with MAX_ROW number of rows of rooms). Room numbers are still between 1 and MAX_ROOM.

```
    public int countEmptyRooms() {

    int count = 0;
    for( int r=1; r<=MAX_ROW; r++) {
        for( int c=1; c<= MAX_ROOM; c++) {
            if ( isFree(r,c)) count++;
        }
    }

    return count;

//OR
    int count = 0;
    for( String[] ns : names) {
        for( String s : ns) {
            if (s == null) count++;
        }
    }
    return count - MAX_ROW - MAX_ROOM - 1;

    //Marker note: answer depends on (b)'s use of rows



Marking:
1: Counter variable
2: Loop all
2: Increment counter for empty rooms
1: Adjust correctly for extra nulls (names [0][...] and names [...][0], etc.)
-.0.5: Minor mistakes
Marking when clearly wrong (max 2):
1: Evidence of learning


    }
```

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