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

EXAMINATIONS – 2022

TRIMESTER 1

COMP 102/112
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 [15]
3. Files [23]
4. ArrayLists of Objects [20]
5. Array and 2D Arrays [35]

June 3, 2022

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) **[6 marks] Calling methods.**

What will the following printStuff method print out?

```
public void printStuff (){
    UI.println("printing stuff");
    int x = 10;
    int y = 20;
    int z = x - y;
    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 = m + n;
    return m;
}
```

Write your answer here

```
printing stuff
z=-10
x=10  y=20  z=30
```

(Question 1 continued on next page)

(Question 1 continued)

Consider the following Member class, specifying objects with two fields, a constructor and three methods. It will be used in 1(b).

```
class Member {
    private String name;
    private double points;

    public Member(String n, double p ){
        this.name = n;
        this.points = p;
    }

    public void event1(){
        this.print();
        this.points = this.points + 10;
    }

    public void event2(double p){
        this.points = this.points - (p / 2);
        this.print();
    }

    public void print(){
        UI.println(this.name + " :" + this.points);
    }
}
```

(Question 1 continued on next page)

(Question 1 continued)

(b) **[7 marks] Calling methods on objects.**

Given the Member class on the facing page, what will the following useObjects method print out?

The boxes on the right are for your working if that helps.

```
public void useObjects(){
    Member m1 = new Member("Thor", 20);
    Member m2 = new Member("Odin", 200);

    UI.println("--------------");
    m1.event1();
    m2.event2(10);
    UI.println("--------------");
    m1.event2(50);
    m2.event1();
    UI.println("--------------");
    m1.print();
    m2.print();
}
```

m1:

m2:

Write your answer here

```
-------------
Thor:  20.0
Odin:  195.0

-------------
Thor:  5.0
Odin:  195.0


-------------
Thor:  5.0
Odin:  205.0
```

(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, 1, 3, 2, 0, 3, 7};

    int n = 1;
    UI.println(nums[5]);
    UI.println(nums[n+1]);
    UI.println(nums.length);
    UI.println("----------");

    for(int i = 1; i <= 3; i++){
        nums[i] = nums[i] + nums[i-1];
        UI.println(nums[i]);
    }
}
```

Write your answer here

```
3
3
7


-------------
6
9
11
```

(Question 1 continued on next page)

(Question 1 continued)

(d) **[7 marks] ArrayLists.**

What will the following arraylistTest method print out?

```
public void arraylistTest (){
    List<Integer> nums = new ArrayList<Integer>();
    nums.add(2);
    nums.add(2);
    nums.add(6);
    nums.add(6);
    nums.add(6);
    nums.add(3);
    nums.add(2);
    nums.add(2);

    int pre_n = nums.get(0);
    int count = 1;
    int num = 0;
    int numCount = 0;

    for(int i = 1; i < nums.size(); i++){
        if (nums.get(i) == pre_n){
            count++;
            if (count > numCount){
                numCount = count;
                num = nums.get(i);
            }
        }
        else{
            count = 1;
        }
        pre_n = nums.get(i);
    }
    UI.println(num);
    UI.println(numCount);
}
```

Write your answer here

6

3

Question 2. Design a class **[15 marks]**

Suppose you are writing a program to draw balls on the screen.

Complete the Ball class on the facing page to specify Ball objects.

- The fields should store the position, the diameter, and the color of a ball. Each ball should have the default start position at (200, 300), where 200 is the x location of the ball centre and 300 is the y position of the ball centre.
- The constructor should create a Ball object with the specified diameter and color. It should call the draw method to show the ball at the default start position.
- The draw method should draw a filled oval for the Ball at the right location using the right color and diameter.
- The erase method should erase a filled oval for the Ball at the right location using diameter. (You can assume that UI.eraseOval(x,y,w,h) erase an oval at position (x,y) with a width of w and height of h.)
- The move(double x, double y, boolean redraw) method should move the ball. The x parameter specifies the number of pixels right. The y parameter specifies the number of pixels down (on the screen). It should erase the old ball and redraw the ball (only) when the redraw parameter is true. Otherwise, nothing should change on the screen.

Write your answer here

```
class Ball {
    private double x = 200;
    private double y = 300;
    private double diameter;
    private Color color;



    public Ball(double d, Color c) {
        this.diameter = d;
        this.color = c;
        this.draw();



    }

    public void draw(){
        UI.setColor(this.color);
        UI.fillOval(this.x-this.diameter/2, this.y-this.diameter,
                    this.diameter, this.diameter);



    }

    public void erase(){
        UI.eraseOval(this.x-this.diameter/2, this.y-this.diameter,
                     this.diameter, this.diameter);



    }
    public void move(double x, double y, boolean redraw){
        if(redraw) this.erase();
        this.x = this.x + x;
        this.y = this.y + y;
        if(redraw) this.draw();



    }
}
```

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

A file is shown below. Each line contains a name and some double numbers. The number of doubles might be different from line to line and the same name may appear in more than one line.

```
Thor 332.5 84.6
Odin 992.3 11.4
Loki 12.0
Freya 43.2 711.2 3.1
Loki 23.5
```

(a) **[8 marks]** What will the following printFromFile method print out if passed the name of the file given above?

```
public void printFromFile(String filename){
    try{
        List<String> lines = Files.readAllLines(Path.of(filename));
        for (int i = 0; i < lines.size(); i++) {
            if(i%2==0) {
                UI.println( lines.get(i));
            }
        }
    } catch(Exception e) {UI.println("Error with file");}
}
```

Write your answer here

```
Thor 332.5 84.6
Loki 12.0
Loki 23.5
```

(b) **[15 marks]** Complete the following personSum method that will read the file and add all the numbers that appear on all the lines starting with the specified name. It will then print the name and the sum.

If the name is not included in the file, show a name + "not found" message.

For instance, if name is assigned Loki, then the methods prints "Loki has 35.5" (12.0 + 23.5). If name is assigned Cthulhu, then the methods prints "Cthulhu not found".

You may use the Files.readAllLines(Path path) method to read all the data from the file, and then process the data line by line. You do not need to add any import statements.

The name of the data file is data.txt.

Write your answer here

```
public void personSum(String name){
    try {
        List<String> lines = Files.readAllLines(Path.of("data.txt"));
        double sum = 0;
        boolean found = false;

        for (String line : lines){

            Scanner s = new Scanner(line);

            if (s.next().equals(name)) {
                found = true;
                while (s.hasNextDouble()) {
                    sum = sum + s.nextDouble();

                }
            }
        }

        if (found) {
            UI.println(name + " has " + sum);
        } else {
            UI.println(name + " not found");
        }
    } catch(Exception e) {UI.println("Error with file");}




}
```

Question 4. ArrayLists of Objects **[20 marks]**

Suppose your family has some big feijoa trees and you are writing a program to help your family to sell the feijoas for $5 per kilo by allowing people to put in orders.

The program has two classes: an OrderManager class and an Order class.

The Order class below defines Order objects.

```
public class Order{
    private String name;
    private double kg;

    public Order(String n, double k) {
        this.name =n;
        this.kg = k;
    }
    public String getName(){
        return this.name;
    }
    public double getKg(){
        return this.kg;
    }
    public void printBill (){
        UI.println(this.name + ": $ " + this.kg * 5);    //$5 per kilo
    }
}
```

(Question 4 continued on next page)

(Question 4 continued)

The OrderManager class declares an orders field to store the list of Order objects:

```
private ArrayList<Order> orders = new ArrayList<Order>();
```

(a) **[7 marks]** What will the following method print out?

```
public void testing(){
    Order a = new Order("Sam", 4);
    Order b = new Order("Lisa", 10);
    Order c = new Order("Alex", 3);
    Order d = new Order("Nick", 5);
    this.orders.clear();
    this.orders.add(a);
    this.orders.add(c);
    this.orders.add(new Order("Tom", 3));
    this.orders.add(0, b);
    this.orders.add(1, d);
    for(Order m: orders){
            UI.println(m.getName());
    }
    this.orders.get(0).printBill();
}
```

Write your answer here

```
Lisa
Nick
Sam
Alex
Tom
Lisa:  $50.0
```

(Question 4 continued on next page)

(Question 4 continued)

(b) **[13 marks]** The following cancelOrder(...) method has two parameters which are the name and the kg.

Complete the cancelOrder(...) method that will search the list of orders to find this particular order with this name and kg, remove it from the list. You may assume there are no duplicate orders on the list.

If this particular order is not found, then print a "not found" message.

Write your answer here

```
public void cancelOrder(String name, double kg) {
    boolean found = false;
    for(int i=0; i<this.orders.size(); i++) {
        if(this.orders.get(i).getName().equals(name) && this.orders.get(i).getKg()==kg) {
            found = true;
            //UI.println("Order for " + this.orders.get(i).getName() + " is removed");
            this.orders.remove(i);

        }
    }
    if (!found) {
        UI.println("Not found");
    }









}
```

Question 5. Array and 2D Arrays **[35 marks]**

This question is about a word search game that searches a word (array of characters) on a deck (array of characters) or a grid (2D array of characters).

Please note that you are required to use character arrays in this questions and you are not allowed to use String methods.

Your method should still work whatever the sizes of the arrays are, as long as they contain at least one value.

The deck and grid are declared as fields and the word is the parameter. You may define your own helper methods.

(a) **[8 marks]** Complete the findFirstChar(char[] word) method to print the indices where the first character of the word is found on the deck. It should print a "not found" message if the first character of the word is not found.

For example, if the deck is declared as follows:

```
private char[] deck = new char[]{'t','g','o','d','o','d','o','g','b','d','d','o','g','d'};
```

The following code:

```
char[] word = new char[]{'d','o','g'};
findFirstChar(word);
```

should print:

```
first char found at 3
first char found at 5
first char found at 9
first char found at 10
first char found at 13
```

Write your answer here

```
public void findFirstChar(char[] word) {
    boolean found = false;
    for(int i=0; i<this.deck.length; i++){
        if(this.deck[i]==word[0]){
            UI.println("first char found at : " + i);
            found = true;
        }
    }
    if (!found){
        UI.println("not found");
    }









}
```

(b) **[15 marks]** Complete the following findWordOnDeck(char[] word) method that will print the indices for all appearances of the word on the deck. It should print a "not found" message if the word is not found.

For example, if the deck is declared as follows: (the same as part a)

```
private char[] deck = new char[]{'t','g','o','d','o','d','o','g','b','d','d','o','g','d'};
```

The following code:

```
char[] word = new char[]{'d','o','g'};
findWordOnDeck(word);
```

should print out the following:

```
word found starting at 5
word found starting at 10
```

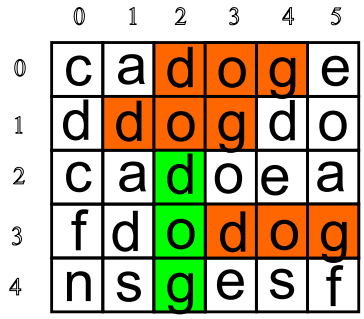
Write your answer here

```
public void findWordOnDeck(char[] word){
public boolean findWordAt(int i, char[] word) {
    for(int k=0; k<word.length; k++){
        if ((i+k>=this.deck.length) ||(this.deck[i+k]!=word[k])){
            return false;
        }
    }
    return true;
}

public void findWordOnDeck(char[] word){
    boolean found = false;
    for(int i =0; i<this.deck.length; i++){
        if (this.findWordAt(i, word)){
            UI.println("word found starting at " + i);
            found = true;
        }
    }
    if (!found) {
        UI.println("not found");
    }
}

// alternative answers
public void findWordOnDeck2(char[] word) {
    boolean found =false;
    for(int j=0; j<this.deck.length; j++){
        if (this.deck[j]==word[0]){
            boolean wholeword=true;
            for(int k=1; k < word.length; k++) {
                if ((j+k>=this.deck.length)||(this.deck[j+k]!=word[k])){
                    wholeword=false;
                    break;
                }
            }
            if (wholeword){
                found =true;
                UI.println("word found starting at :" + j );
}           }
        }
    }
    if (!found){
        UI.println("not found");
    }
}
```

(c) **[12 marks]** Complete the findWordInGrid(char[] word) method to search for a word in a 2D array of characters, and print the starting locations as "found starting at (row index, col index)" for each appearance of the word. It should print a "not found" message if the word is not found.

The grid field is declared to hold a 2D array of characters, for example

```
private char [][] grid = new char[][]{
        {'c','a','d','o','g','e'},
        {'d','d','o','g','d','o'},
        {'c','a','d','o','e','a'},
        {'f','d','o','d','o','g'},
        {'n','s','g','e','s','f'}};
```



The following code

```
char[] word = new char[]{'d','o','g'};
findWordInGrid(word)
```

should print out the following:

```
found starting at (0, 2)
found starting at (1, 1)
found starting at (2, 2)
found starting at (3, 3)
```

Please note that we are searching in **2** directions rather than 1 or 4. Your method should be able to search both **horizontally (from left to right)** and **vertically (from top to bottom)**. In the above example, (2, 2) is found vertically.

```
public void findWordInGrid(char[] word) {


    public boolean findWordInRowAt(int i, int j, char[] word) {
        for(int k=0; k < word.length; k++) {
            if ((j+k>=this.grid[i].length )||( this.grid[i][j+k]!=word[k])){
                return false;
            }
        }
        return true;
    }

    public boolean findWordInCollumnAt(int i, int j, char[] word) {
        for(int k=0; k < word.length; k++) {
            if ((i+k>=this.grid.length )||( this.grid[i+k][j]!=word[k])){
                return false;
            }
        }
        return true;
    }
    public void findWordInGrid(char[] word) {
        boolean found = false;
        for (int i=0; i<this.grid.length; i++){
            for (int j=0; j < this.grid[0].length; j++){
                if (this.grid[i][j]==word[0]) {
                    if (this.findWordInRowAt(i,j, word)){
                        found =true;
                        UI.println("found horizontally at (" + i +", " + j +")");
                    }
                    if (this.findWordInCollumnAt(i,j,word)) {
                        found =true;
                        UI.println("found vertically at (" + i +", " + j +")");
                    }
                }
            }
        }
        if (!found){
            UI.println("not found");
        }
}
    }
```

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.

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.

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