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

EXAMINATIONS – 2019

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. 2D Arrays [20]
5. ArrayLists of Objects [35]

July 7, 2020

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 = 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;
}
```

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

(Question 1 continued on next page)

(Question 1 continued)

Consider the following Member class, specifying objects with two fields, a constructor and three methods.

```
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 + 5;
    }

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

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

(Question 1 continued on next page)

(Question 1 continued)

(b) **[6 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("Alex", 20);
    Member m2 = new Member("Bob", 200);

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

m1:

m2:

Write your answer here

```
Alex:  20.0
Bob:  150.0

-------------
Alex:  15.0
Bob:  150.0


-------------
Alex:  15.0
Bob:  155.0
```

(Question 1 continued on next page)

(Question 1 continued)

(c) **[8 marks] Arrays.**

What will the following arrayTest method print out?

```
public void arrayTest(){
    int [ ] nums = new int [ ]{2, 6, 3, 4, 1, 2, 9, 0};

    int n = 0;
    UI.println(nums[3]);
    UI.println(nums[n+1]);
    UI.println(nums.length);
    UI.println("-----------");

    for(int i = 0; i < 3; i++){
        nums[i] = nums[i] + nums[i+1];
        UI.println(nums[i]);
    }
}
```

```
4
6
8

-------------
8
9
7
```

(Question 1 continued on next page)

(Question 1 continued)

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

Complete the following countTHE method that will count and print the number of times that the word "THE" appears in the ArrayList of Strings stored in words.

For example, if the user enters "The cat sat on the mat", it should print "THE appeared 2 times"

You method should not be case sensitive.

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









    int count = 0;
    for (String s: words){
        if (s.equalsIgnoreCase("THE")){
            count++;
        }
    }
    UI.println("THE appeared " + count + " times");




}
```

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

Suppose you are writing a program to simulate a production line where many boxes are moving on a belt from left to right.

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

- The fields should store the position, the width, the height, and the color of a box. Each box should have the default start position at (100, 300), where 100 is the centre of the x location of the box and 300 is the y position of the belt.
- The constructor should create a Box object with the specified height, width and color. It should call the draw method to show the box at the default start position.
- The draw method should draw a filled rectangle for the Box at the right location using the right color and size.
- The move(double pixels) method should move the box to the right for the specified number of pixels. It should not redraw it.
- The maxSize method should return the greater number of its height and width.

```
public class Box {


    private double x = 100;
    private double y = 300;
    private double width;
    private double height;
    private Color color;
    public Box(double w, double h, Color c) {



        this.width = w;
        this.height = h;
        this.color = c;
        this.draw();
    }
    public void draw(){




        UI.setColor(this.color);
        UI.fillRect(this.x-this.width/2, this.y-this.height, this.width, this.height);



    }
    public void move(double s){


        this.x = this.x + s;

    }
    public double maxSize(){


        if (this.width > this.height)
            return this.width;
        return this.height;
    }
}
```

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

A file is shown below. Each line contains a year and some integer numbers. The number of integers might be different from line to line and there might be multiple lines for one year.

```
2019 3 4 6
2017 3 8
2018 9 1
2015 1 7 9 10 4 2 1
2017 4 7 3
2019 1 4 3
2019 3 4 5 8
```

(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){
    ArrayList<String> lines = readAllLines(filename);
    for (int i = 4; i <= 6; i++) {
        UI.println(lines.get(i));
    }
}
public ArrayList<String> readAllLines(String fname){
    ArrayList<String> ans = new ArrayList<String>();
    try {
        Scanner scan = new Scanner(new File(fname));
        while (scan.hasNext()){
            ans.add(scan.nextLine());
        }
        scan.close();
    }
    catch(IOException e){UI.println("File reading failed");}
    return ans;
}
```

```
2017 4 7 3
2019 1 4 3
2019 3 4 5 8
```

(b) **[15 marks]** Complete the following yearAverage method that will read the file, find all the numbers for one particular year specified in the parameter, calculate and print the average of the numbers.

If the year is not included in the file, show a "not found" massage.

You may call the readAllLines method on the facing page to read all the data from the file, and then process the data line by line.

The name of the data file is data.txt.

```
public void yearAverage(int year){

    ArrayList<String> lines = readAllLines("data.txt");
    double sum = 0;
    int count = 0;
    for (String line : lines){

        Scanner s = new Scanner(line);
        if (s.nextInt() == year){
            while (s.hasNextInt()) {
                sum = sum + s.nextInt();
                count++;
            }
        }
    }
    if (count > 0)
        UI.println(year + ": " + sum/count);
    else
        UI.println("not found");









}
```

Question 4. 2D Arrays **[20 marks]**

The data field is declared to hold a 2D array as follows:

```
private double[ ][ ] data = new double[ ][ ]{{11.44, 39.8,  75.9, 12},
                                             {44.95, 39,    83,   30},
                                             {49.38, 49.9,  7.5,  90},
                                             {47.18, 14.61, 98.7, 91},
                                             {14.27, 78.9,  8,    17.6}};
```

(a) **[8 marks]** What will the following method print out?

```
public void printNums(){
    UI.println(data[0][3]);
    for (int i = 0; i < 2; i++){
        for(int j = 0; j < 2; j++){
            UI.println(i + ": "+ data[i][j]);
        }
    }
}
```

```
12
0:  11.44
0:  39.8
1:  44.95
1:  39.0
```

(Question 4 continued on next page)

(Question 4 continued)

(b) **[12 marks]** Complete the following findRowMaxs method that will find and print the maximum number on each row.

For example, if the 2D array contains the data in part (a), findRowMaxs should print out the following:

```
0: 75.90
1: 83.00
2: 90.00
3: 98.70
4: 78.90
```

Your method should still work whatever the size of the array is, assuming that every row of the array contains at least one column.

```
public void findRowMaxs(){

    for (int i = 0; i<data.length; i++){
        double max = data[i][0];
        for(int j =0; j<data[i].length; j++){
            if(data[i][j]>max){
                max = data[i][j];
            }
        }
        UI.printf("%d: %.2f\n", i, max);
    }









}
```

Question 5. ArrayLists of Objects **[35 marks]**

This question concerns a program for estimating the polarity score of a sentence using a dictionary. For example, "excellent movie, great performance" should have a positive score, and "bad movie" should have a negative score.

The program has two classes: a SentenceEvaluator class and a PolarityWord class.

The PolarityWord class below defines PolarityWord objects, which store each word with its polarity, for example, "excellent" has a polarity of 3; "good" has a polarity of 2, and "bad" has a polarity of -2.

```
class PolarityWord {
    private String word;
    private int polarity ;


    public PolarityWord(String w, int p) {
        this.word = w;
        this. polarity = p;
    }

    public String getWord(){
        return this.word;
    }

    public int getPolarity (){
        return this. polarity ;
    }
}
```

(Question 5 continued on next page)

(Question 5 continued)

The SentenceEvaluator class declares a dictionary field to store the list of PolarityWord objects:

```
private ArrayList<PolarityWord> dictionary = new ArrayList<PolarityWord>();
```

(a) **[8 marks]** What will the following method print out?

```
public void testArrayList (){
    PolarityWord a = new PolarityWord("excellent", 3);
    PolarityWord b = new PolarityWord("good", 2);
    PolarityWord c = new PolarityWord("fine", 1);
    PolarityWord d = new PolarityWord("bad", -2);
    PolarityWord e = new PolarityWord("terrible", -3);

    dictionary.clear();
    dictionary.add(a);
    dictionary.add(c);
    dictionary.add(new PolarityWord("poor", -1));
    dictionary.add(new PolarityWord("awesome", 3));
    UI.println(dictionary.get(0).getWord());
    UI.println("-----------------");
    UI.println(dictionary.size());
    UI.println(dictionary.get(1).getPolarity());
    dictionary.set(2,d);
    UI.println("--------------------");
    for(PolarityWord f: dictionary){
        UI.println(f.getWord());
    }
}
```

```
excellent

-----------------
4
1

---------------
excellent
fine
bad
awesome
```

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.

(b) **[12 marks]** Suppose we have created a dictionary with some important words and their polarity. We want to use this dictionary to estimate the polarity score of a sentence.

The following scoreSentence method has a parameter which is the sentence entered by the user and you may assume there is no punctuation in it.

Complete the scoreSentence method that will search the dictionary to find the polarity of each word, calculate and return the score of the sentence as the sum of all word polarity scores. If a word is not in the dictionary, its polarity score is zero.

Your method should not be case sensitive, so that "Good", "GOOD" can match with "good" in the dictionary.

Hint: It is easier to do this in two steps. First define a method to search and return the polarity of a word (this method is also useful for the next sub-question), and then define a method to calculate the score of a sentence.

```
public int scoreSentence(String sentence){



    Scanner s = new Scanner(sentence);
    int score = 0;
    while (s.hasNext()){
        String word = s.next();
        UI.println(word);
        score = score + scoreWord(word);
    }
    return score;




}
public int scoreWord(String word){
    int score = 0;
    for (PolarityWord w: dictionary){
        if (w.getWord().equalsIgnoreCase(word)){
            score = w.getPolarity();
        }
    }
    return score;









}
```

(c) **[15 marks]** Complete the following ScoreSentenceWithNegation method that will compute and return the score of a sentence taking into account when "not" is used. If the word "not" is found in a sentence, *the next "polarity word" that is close enough*, will change its score from positive to negative, or vice versa.

By *the next "polarity word" that is close enough*, we mean a word with a non-zero polarity score, that is after the word "not" and not more than 5 words away.

For example, in "not too bad", the next polarity word is "bad", and its score changes from -2 to 2. However, in "not the right kind of movie for me It is too bad", the word "bad" is too far away, so its polarity score is not changed.

You can use the scoreWord method you defined in (b).

```
public int scoreSentenceWithNegation(String sentence){



    Scanner s = new Scanner(sentence);
    int score = 0;
    int wordScore =0;
    while (s.hasNext()){
            String word = s.next();
            if (word.equalsIgnoreCase("not")){
                    int count = 0;
                    while (s.hasNext()){
                            count++;
                            if (count > 5){
                                    break;
                            }
                            String nextWord = s.next();
                            wordScore = scoreWord(nextWord);
                            if (wordScore!=0){
                                    wordScore = -1 * wordScore;
                                    UI.println(wordScore);
                                    break;
                            }
                    }
            }
            else{
                    wordScore = scoreWord(word);
            }
            score = score + wordScore;
    }
    return score;









}
```

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