GRIFFITH COLLEGE DUBLIN
GRIFFITH COLLEGE CORK
GRIFFITH COLLEGE LIMERICK

QUALITY AND QUALIFICATIONS IRELAND

EXAMINATION

BACHELOR OF SCIENCE IN COMPUTING

STAGE I

COMPUTER PROGRAMMING
Module code: BSCO-CP

BACHELOR OF SCIENCE (HONS) IN COMPUTING SCIENCE

STAGE I

COMPUTER PROGRAMMING
Module code: BSCH-CP

Lecturer(s): **Tracey Cassells**

Tianshu Xu

Martin Dow

External Examiner(s):

Date: August 2023 **Time:**

THIS PAPER CONSISTS OF SIX QUESTIONS

FIVE QUESTIONS TO BE ATTEMPTED

SECTION A – COMPULSORY

SECTION B – FOUR QUESTIONS TO BE ATTEMPTED

SECTION A – COMPULSORY

QUESTION 1

a) Write Boolean expressions for the following (where A, B, C, D, E, F, G are integers):

i. A is a multiple of both B and C.

((A%B == 0) && (A%C == 0)) **(2 marks)**

ii. D is either negative or greater than 100.

((D < 0) || (D > 100)) **(2 marks)**

iii. A is divisible by 4, and either not divisible by 100 or divisible by 400.

((A % 4 == 0) && ((A % 100 != 0) || (A % 400 == 0))) **(2 marks)**

iv. The sum of A and F is less than or equal to the product of B and C.

((A + F) <= (B + C)) **(2 marks)**

v. G is a positive even number that is not divisible by 3.

((G > 0) && (G % 2 == 0) && (G % 3 != 0)) **(2 marks)**

(10 marks)

b) Please explain the difference between the following 2 statements.

```
int x = 5;

x = 6;
```

```
int x = 10;

x == 10;
```

In your answer highlight any issues you can see with either of the two statements.

The first box creates an int called x and initialises it to 5, then changes its value to 6. **(2 marks)**

The second box creates an int called x and initialises it to 10, then checks to see if the value of x is equal to 10 **(3 marks)**

(5 marks)

c) Explain the difference between a private and public access modifier in java. Why would we make a variable private?

When a method or variable is declared as "public", it can be accessed by any part of your code, regardless of the package or class it belongs to. **(2 marks)**
Private methods and variables are not visible to any other classes or parts of your program, which can be useful for encapsulating the implementation details of your code**. (2 marks)**
You would use private variables to store data that should not be modified by other parts of your code**. (1 mark)**

d) The library method Math.random() yields a real number in the range

$$0.0 \leq \text{Math.random()} < 1.0$$

Using this method, write down a Java expression denoting a random integer in the range 1..1024 inclusive.

```
(int) ( Math.random( ) * 1024 ) + 1;
```

$$0 \leq \text{(int) (Math.random( ) * 1024)} \leq 1023$$

$$1 \leq \text{(int) (Math.random( ) * 1024) + 1} \leq 1024$$

Correct minimum number. **(2 marks)**

Correct maximum number. **(2 marks)**

Answer looks right in the end. **(1 marks)**

(5 marks)

e) Study the following code section carefully and answer the question below.

```
public static void main (String[] args) {
    int x = 5;
    double y = x / 2;
```

```
    System.out.println(y);
}
```

What is the output of this code?.

(5 marks)

Marks (5)

- 2.5 **(1 mark)** (double) x / 2
- 2 **(1 mark)** (int) (x / 2)
- 2.0 **(3 marks)** correct

f) Study the following code section carefully and complete the task below.

```
public static void main(String[] args) {
// a, b, and c are sides of a right triangle, c is the hypotenuse
        double a = 15.5;
        double b = 12;
        double c = Math.sqrt(a*a + b*b);
        System.out.println(c);
    }
```

Your task is to rewrite the code as a method called calculateHypotenuse that takes a and b sides as arguments and returns the calculated c side. Call calculateHypotenuse from the main method with the values given in the code section and print the returned value.

(5 marks)

Marks (5)

- public static double calculateHypotenuse (double a, double b) **(2 marks)**
- return calculated value **(1 mark)**
- call method from main and print result **(2 marks)**

Solution:

```
public static void main(String[] args)
{
        System.out.println(calculateHypotenuse(15.5,12));
}


public static double calculateHypotenuse(double a,double b)
{
         return Math.sqrt(a*a + b*b);
}
```

g) Given the character array,

char[] letters = {'g','t','A','b','Z'};

Write the code to do the following:

1. Loop through the array and print each character. **(2 marks)**
2. Print out any lowercase characters. **(2 marks)**
3. Print out all characters with a Unicode value lower than 'a'. **(1 mark)**

Total (5 marks)

Marks (5)

Any looping structure **(1 mark)**
Access each element in letters **(1 mark)**
Check if lowercase **(2 marks)**
Check if less than a **(1 mark)**

Solution:

```
for( char letter : letters)
{
        System.out.println(letter);
        if( Character.isLowerCase(letter))
                System.out.println(letter);
       if(letter < 'a')
```

```
                System.out.println(letter);
        }
```

Total (40 marks)

SECTION B – FOUR QUESTIONS TO BE ATTEMPTED

QUESTION 2

Write a method called letterFrequency that takes a character and an array of Strings as an argument and returns the word that contains the most instances of the given character.
The method should ignore case.

Example input: 'e' , {"hello there", "nice weather today ", "I overslept yesterday", "Evening everyone" }

Example returned value: { "Evening everyone"}

(15 marks)

Marks (15)

- public static String letterFrequency(char c, String[] strings) **(2 marks)**
- loop through each string in strings **(2 marks)**
- loop through each character in the string **(2 marks)**
- count how many times the character appears. **(3 marks)**
- get the String with the maximum frequency **(2 marks)**
- return the string **(2 marks)**

- ignore case **(2 marks)**

Solution:

```
public static String letterFrequency(char c, String[] strings) {
   String result = "";
   int max = 0;

   for (String line : strings) {
      int count = 0;
      for (int i = 0; i < line.length(); i++) {
         if (Character.toLowerCase(line.charAt(i)) == Character.toLowerCase(c)) {
            count++;
         }
      }
      if (count > max) {
         max = count;
         result = line;
      }
   }
   return result;
}
```

QUESTION 3

```
char[][] board1 = {
                { 'R','N','B','Q','K','B','N','R'},
                { 'P','P','P','P','P','P','P','P'},
                { ' ',' ',' ',' ',' ',' ',' ',' '},
                { ' ',' ',' ',' ',' ',' ',' ',' '},
                { ' ',' ',' ',' ',' ',' ',' ',' '},
                { ' ',' ',' ',' ',' ',' ',' ',' '},
                { 'p','p','p','p','p','p','p','p'},
                { 'r','n','b','q','k','b','n','r'}
};

char[][] board2 = {
                { ' ',' ',' ',' ',' ',' ',' ',' '},
                { ' ',' ','r',' ',' ',' ',' ',' '},
                { ' ',' ',' ','n',' ',' ',' ',' '},
                { ' ',' ',' ',' ',' ',' ','K',' '},
                { ' ',' ',' ',' ',' ',' ',' ','R'},
                { ' ',' ',' ',' ',' ',' ',' ',' '},
                { ' ',' ',' ',' ',' ',' ',' ',' '},
                { ' ',' ',' ',' ',' ',' ',' ',' '}
};
```

Black pieces	
K	King
Q	Queen
R	Rook
B	Bishop
N	kNight
P	Pawn
White pieces	
k	King
q	Queen
r	Rook
b	Bishop
n	kNight
p	Pawn
' '	No piece

A chess game board is represented by a 2D array of type char in Java. Each location in the array is either represented by the char corresponding to the piece as shown in the array declarations above, or the space character (' ') if no piece is present at the square.

Your task is to write a method called isGameOver which determines whether there is a winner and the game is over by whether a king has been removed from the board. The isGameOver method should return a boolean. isGameOver would return false for board1 above, and true for board2. The isGameOver method is passed a 2D array of char to represent the board to test.

Ensure your code can cater for alternative rectangular boards other than the well-known 8x8 – for example, 6x5 or 4x4.

Write code to create a small test board of dimensions 2x3 and show how to pass it to the isGameOver method.

(15 marks)

Marks: 15

Correctly instantiate method with int array argument **(1)**
Outer loop **(2)**

Inner loop **(2)**
Conditional **(2)**
Temporary variable when kings are encountered **(2)**
Create test board **(2)**
Call isGameOver correctly **(2)**
Checks for array dimensions within code (no magic numbers) **(2)**
Create test board and pass to isGameOver method **(2)**

Solution:

```
public class Q3 {


	private static boolean isGameOver( char[][] board ) {
		int kingsFound = 0;
		for( int row=0; row < board.length; row++ ) {
			for( int col=0; col < board[row].length; col++ ) {
				if( board[row][col] == 'K' || board[row][col] == 'k' )
					// return as soon as two kings found
					if(++kingsFound == 2)
						return false;	// ##### RETURN ######
			}
		}
		return true;			// ##### RETURN ######
	}


	public static void main(String[] args) {


		char[][] testBoard = {
				{'k','Q',' '},
				{' ',' ','q'}
		};
		System.out.println("oddBoard: " + isGameOver(testBoard));
```

```
    }
}
```

QUESTION 4

Create a method and name it "insertionSort". The method should take an integer array as an input parameter. Use the Insertion Sort algorithm to sort the given integer array in ascending order. The method does not return anything but prints the sorted array on screen.

(15 marks)

Marks: 15

- public static void insertionSort(int[] arr) **(3 marks)**
- Outer loop **(2 marks)**
- Inner loop **(2 marks)**
- Compare values **(3 marks)**
- swap values using a temporary value **(3 marks)**
- print the sorted array **(2 marks)**

Solution:

```
static void insertionSort(int[] arr)
{
    for (int j = 1 ; j < arr.length; j++)
    {
        int temp = arr[j];
        for (int i = j; i > 0 && arr[i] < arr[i-1];i--)
        {
            arr[i] = arr[i-1];
            arr[i-1] = temp;
        }
```

```
        }
    System.out.println("Sorted array: " + Arrays.toString(arr));
}
```

QUESTION 5

Write a Java program that reads a text file called **chapters.txt** and write another text file called **sorted.txt**. Each line of the input file consists of a 1 to 10 unique chapter number and a chapter title separated by a comma, e.g.,

1, Back to normal
3, Forth thy soul
2, And so on
5, Forever

The program should write the sorted chapters to the sorted.txt file, e.g.,

1, Back to normal
2, And so on
3, Forth thy soul …

Ensure you appropriately deal with FileNotFoundException and IOException. You do not need to include imports in your answer

(15 marks)

Marks: 15

- Open chapters.txt **(2 marks)**
- Read each line (chapter) **(2 marks)**
- Add the chapters to a list/array **(2 marks)**
- Sort the chapters (may split with comma) **(4 marks)**
- Append to sorted.txt **(2 marks)**
- Close readers and writers **(1 marks)**
- Catch exceptions **(2 marks)**

Solution: *(other File classes accepted)*

```
try {
        Scanner read      = new Scanner(new File("chapters.txt"));
        FileWriter write = new FileWriter("sorted.txt");

        ArrayList<String> chapters = new ArrayList<>();
        while(read.hasNextLine())
        {
            String chapter = read.nextLine(); // "1, Back to normal"
            chapters.add(chapter);
        }
        Collections.sort(chapters);
        for    (String str: chapters)
            write.write(str+"\n");
        read.close();
        write.close();
    } catch (FileNotFoundException e) {
        System.out.print("Open the File: Failed");
    } catch (IOException e) {
        System.out.print("Write  to File: Failed");
    }
```

QUESTION 6

Write a class called Engine. This class should have four private variables: `manufacturer` (type String), `modelName` (type String), `powerKiloWatts` (type int, representing engine power in kiloWatts as a whole number). The class should also have a constructor which initialises all variables. Your code must check `powerKiloWatts` is positive; if it is not then it is assigned the value 0 (zero). The class must include a complete set of accessor (getter/setter) methods for all variables.

Write a public method `getPowerHP` which obtains the power in horse-power using the formula 1 Watt = 1359.6216173 horsepower (metric) and returns a double.

Write code to create an example Engine object and call `getPowerHP` on the created engine instance. Print the manufacturer, model and power in horsepower to the console in your test code.

(15 marks)

Marks: 15

Correct class construction **(2)**
Use private member variables **(1)**

Correct constructor **(2)**
Check for positive int powerKiloWatts argument within constructor **(1)**
Getters x 3, each **(0.5)**
Setters x 3, each **(0.5)**
Check for positive int arguments within setPowerKiloWatts setter **(1)**
Correct conversion method to horsepower getPowerHP **(2)**
Create at least one test engine **(1)**
Call getPowerHP **(1)**
Print result to the console **(1)**

Solution:

```
public class Engine {

	private String manufacturer;
	private String modelName;
	private int powerKiloWatts;

	public Engine(String manufacturer, String modelName, int powerKiloWatts) {
		this.manufacturer = manufacturer;
		this.modelName = modelName;
		this.powerKiloWatts = powerKiloWatts < 0 ? 0 : powerKiloWatts;
	}

	public String getManufacturer() {
		return manufacturer;
	}
	public void setManufacturer(String manufacturer) {
		this.manufacturer = manufacturer;
	}
	public String getModelName() {
		return modelName;
	}
	public void setModelName(String modelName) {
		this.modelName = modelName;
	}
	public int getPowerKiloWatts() {
		return powerKiloWatts;
	}
	public void setPowerKiloWatts(int powerKiloWatts) {
		this.powerKiloWatts = powerKiloWatts < 0 ? 0 : powerKiloWatts;
	}
	public double getPowerHP() {
		return powerKiloWatts * 1.3596216173;
```

```
	}

	public static Engine[] makeSampleEngines() {
		return new Engine[] {
			new Engine( "Rolls Royce", "Phantom V12", 350),
			new Engine( "Rolls Royce", "Wraith V12", -320),
			new Engine( "Rolls Royce", "Ghost V12", 320)
		};
	}

	public static void main(String[] args) {
		for (Engine e : makeSampleEngines())
			System.out.println(e.getManufacturer() + "\t"
					+ e.getModelName() + "\t"
					+ e.getPowerHP() + " Horsepower (metric)");

	}
}
```