Practical-Exam-02: Java and Object-Oriented Programming

Due 1 Apr by 16:55 **Points** 100 **Available** 1 Apr at 15:00 - 1 Apr at 17:00 2 hours

This assignment was locked 1 Apr at 17:00.

Introduction

This session ends 4:55 pm (WR02, WR03)

Make sure to complete your submission in time.

Submit to SVN & Websubmission (for every problem solved)

Submission

- Create the repository on the SVN server
- You must checkout only the folder practical-exam-02 from your server
- No other folders from your SVN will be allowed during the practical exam.
- Check if your repository is being track by WebSubmission before start solving the exam.

```
svn mkdir -m "first assessment commit" --parents https://version-control.adelaide.edu.au/svn/<
YOUR-ID>/2022/s1/fcs/week-05/practical-exam-02
```

Assessment

- **This is a practical exam - your work must be entirely your own.**
- Marks for this practical exam contribute 2 marks.
- Marks will be awarded later by your workshop supervisor (30%) and websubmission marker (70%).
- **Due date: 5 minutes before end of lab session**
- **Do Not Forget** To Submit on **WebSubmission (https://cs.adelaide.edu.au/services/websubmission/?sub_year=2020)**
- **Late penalties:** Only work submitted during your enrolled practical session from a Linux system in the practical lab will be marked.

Regarding functional marks, please consider:

```
(1) only codes that can compile will be marked;
(2) only codes that are in the suggested directory will be marked;
(3) only codes submitted to SVN and WebSubmission before the deadline will be marked;
(4) only codes containing your signature on the top of the file will be marked by tutors;
(5) you will have your markers decreased in 3 points if *.class file present in your folder
s;
```

Signature on your files

Note that all your coding files must contain on the top of it this information:

```
//==================================
// Foundations of Computer Science
// Student: you name
// id: your id
// Semester:
// Year:
// Practical Exam Number:
//===================================
```

Note

- To acquire full marks **(1)** all your functionalities must be working perfectly, **(2)** your code has to be well and proportionally commented, and **(3)** your code must follow correct indentation (4 spaces for every new indentation level) and **(4)** you have to use all the content from latest lectures.
- We argue that you are not just asked to solve a problem but use the more sophisticated way to solve it. For instance, you can solve a problem using ten variables, but it will always be better to solve the same problem with an array.

Practical Exam 02

Part 01 - Basic Programming using Java

Problem 01 - Functions Sum, Cube and Division

Define and implement the functions sum, cube and division that have the following signatures:

```
Signature:

public class Calculator

public int sum(int numA, int numB);
// add two integer numbers

public float cube(float num);
// calculate num^3, for example  cube(2.0) returns 8.0

public float division(float numA, float numB);
// return   numA/numB
// If division by 0, the result should be -99.9;
```

```
Repository

save this project as practical-exam-02/problem-01/Calculator.java

add to your svn repository

Important

1. The class Calculator MUST NOT contain a public static main function.
2. If you want to test your code, create an extra class to test it. This
   class should not be submitted to your svn repository;

Constraints
You are not asked to print any information on the screen. The
testing scripts will perform this task
```

Problem 02 - Debugging

A runtime error is a program error that occurs while the program is running. The term is often used in contrast to other types of program errors, such as syntax errors and compile-time errors. There are many different types of runtime errors. One example is a logic error, which produces the wrong output.

There is a RunTime issue in the bugMethod code and a compile-time issue in the bugMethod2. You are asked to fix the following code, and submit to your SVN.

```
//Code:

public class DebuggingDemo {

    public void bugMethod() {
         int num[] = {1, 2, 3, 4};
         System.out.println(num[5]);
     }

    public int bugMethod2() {
            float ans = 0.0;
            System.out.println("This method had a bug!");
            return ans;
     }


}
```

```
Current output:

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5

at Exceptions.Unchecked_Demo.main(Unchecked_Demo.java:8)
```

```
Repository

save this project as practical-exam-02/problem-02/DebuggingDemo.java

add to your svn repository

Requirements

1. Fix the bugs;
2. Provide comments explaining what caused the bug;
3. The classes MUST NOT contain a public static main function.
```

Problem 03 - Handling Arrays

Define and implement a code to **Handle Arrays**.

```
public class HandlingArrays {

    public static void printArray(double [] testArray) {
          // your code goes here
     }

    public static double[] sumElements(double [] firstArray, double [] secondArray) {
          // your code goes here
     }

    public static double[] maxArrays(double [] firstArray, double [] secondArray) {
          // your code goes here
     }

     public static double[] maxAnyArrays(double [] firstArray, double [] secondArray) {
          // your code goes here
     }
}
```

You are asked to define:

- Define and implement the `printArray`, for a given array;
- Define and implement `sumElements`, which sum **element-wise** two arrays; The term element-wise means that you want to perform an operation on each element of a vector while doing a computation. For example, you may want to add two vectors by adding all of the corresponding elements. You might use java.lang.RuntimeException **[link] (https://docs.oracle.com/javase/7/docs/api/java/lang/RuntimeException.html)** ;

```
Quick tip:

if( // condition checking whether an array is bigger than the other ){
                throw new RuntimeException("Error - Arrays different shape");
}else{
 // do something
}
```

- Define and implement the `maxArrays`, which compares **element-wise** two arrays of the same

- Define and implement the `maxArrays`, which compares **element-wise** two arrays of the same size, and returns an array with the largest values in each position. See output expected for examples
- Define and implement the `maxAnyArrays`, which is similar to maxArrays but allows to have arrays of different sizes, by assuming the missing elements are smaller.

```
Output Expected:
PrintArray method:
Given array {1.0,2.0,3.0,4.0}, for instance. The output of printArray should be:
[1.0,2.0,3.0,4.0]

sumElements method:
Given array {1.0, 2.0, 3.0} and {3.0, 4.0, 6.0}, sumElements should return the array
{4.0, 6.0, 9.0}
Given two arrays of different size, sumElements should rise an exception that outputs:
Error - Arrays different shape.


maxArrays method:
Given array {1.0, 3.0, 5.0} and {3.0, 4.0, 2.0}, maxArrays should return the array
 {3.0, 4.0, 5.0}
Given array {5.2, 8.2, 0.5, 22.4} and {3.5, 9.0, 2.0, 20.8}, maxArrays should return the array
 {5.2, 9.0, 2.0, 22.4}
Given two arrays of different size, maxArrays should rise an exception that outputs:
Error - Arrays different shape.

maxAnyArrays method:
Given array {1.0, 5.0, 3.0} and {3.0, 4.0, 6.0}, maxAnyArrays should return the array
{3.0, 5.0, 6.0}
Given array {1.0, 5.0, 3.0} and {3.0, 4.0, 6.0, 7.0}, maxAnyArrays should return the array
{3.0, 5.0, 6.0, 7.0}
Given array {5.0, 5.0, 5.0, 5.0} and {3.0, 7.0}, maxAnyArrays should return the array
{5.0, 7.0, 5.0, 5.0}


Important
1. The class HandlingArrays MUST NOT contain a public static main function.
2. In order to test your code prior to submission, please create another class Main.java and pe
test from that class.
3. The method printArray display information on the screen. The other methods just return an ar

Repository
save this project as practical-exam-02/problem-03/HandlingArrays.java
add to your svn repository
```

```
The classes MUST NOT contain a public static main function.
```

Part 02 - Object-Oriented Programming

Problem 04 - Creating Classes, Objects, Accessors, and Mutator

In this problem, you are required to create a few basic classes and set accessors and mutators for their attributes.

```
// filename: Character.java
public class Character
+ properties:
private:
+++ (String) name; //this attribute storage the character name;
+++ (int) age; // this attribute storage the character age;
+++ (String) gender; // this attribute storage the character gender
+++ (String) occupation; // this attribute storage the character occupation
+++ (String) familyRole; // this attribute storage the family role
+++ (float) rate;  // this attribute storage the character overall rate by fans;

+ methods:
public:
++ Constructor: create (1) default constructor, and (2) a Constructor containing all parameter
s;
++ accessors: for all the attributes;
++ mutator: for all the attributes;


// filename: Cake.java
public class Cake
+ properties:
private:
+++ (String) name;
+++ (float) qtSugar;
+++ (float) qtFlour
+++ (float) qtYeast;
+++ (double) timePrepare;

+ methods:
public:
++ Constructor: create (1) default constructor, and (2) a Constructor containing all parameter
s;
++ accessors: for all the attributes;
++ mutator: for all the attributes;


// filename: Car.java
public class Car
+ properties:
private:
+++ (String) model;
+++ (int) numGears;
+++ (float) literTank;
+++ (int) yearManufacture;
+++ (int) mileage;

+ methods:
public:
++ Constructor: create (1) default constructor, and (2) a Constructor containing all parameter
s;
++ accessors: for all the attributes;
++ mutator: for all the attributes;
```

''''

Repository

save this project as practical-exam-02/**problem-04/*.java**
Character.java, Cake.java and Car.java

add to your svn repository

Requirements

1. Create classes;
2. Create Constructors;

```
3. Create Accessors and Mutators;
```

```
The classes MUST NOT contain a public static main function.
```

Basic Marking Scheme		
Criteria	**Ratings**	**Pts**
Compilation In order to achieve full marks - your code must compile and run;		20 pts
Basic Functionality Your code (1) perform all the functions correctly, (2) use latest concepts learned in class, (3) has a clear, creative and sophisticated way of solving the problem.		40 pts
Functionality Extension Your code (1) perform all the functions correctly, (2) use latest concepts learned in class, (3) has a clear, creative and sophisticated way of solving the problem, and (4) you propose novel ways to solve the problems - or extra functionalities.		10 pts
Code Formatting Your code (1) has the right proportion of comments and place line and block comments correctly, (2) follow correct indentation every new indentation level, (3) has good variable naming, (4) has clear organization between tabs and is easy to read.		30 pts
		Total points: 100