CEGEP VANIER COLLEGE
CENTRE FOR CONTINUING EDUCATION
Programming Algorithms and Patterns
420-930-VA

Teacher: Samir Chebbine **Lab 1** **May 21, 2024**

Lab 1: Arrays, Arrays of objects and ArrayList

Complete all these following programs as explained during **classes.** All *missing coding statements* were provided there with explanation. Create and Submit a Word file ***Lab1OOPProgramminAlgorithmsYourName.docs*** which includes output screenshots for every Java Project. Submit the Java projects too.

1. Data Structure

a) Reading Data From Text File without Array

Create a Java Project to be named ***FileProcessProject*** using Eclipse IDE for reading input file *divide.in*. Create *TestFileProcess1.java* to read from input file as shown in Figure 1.

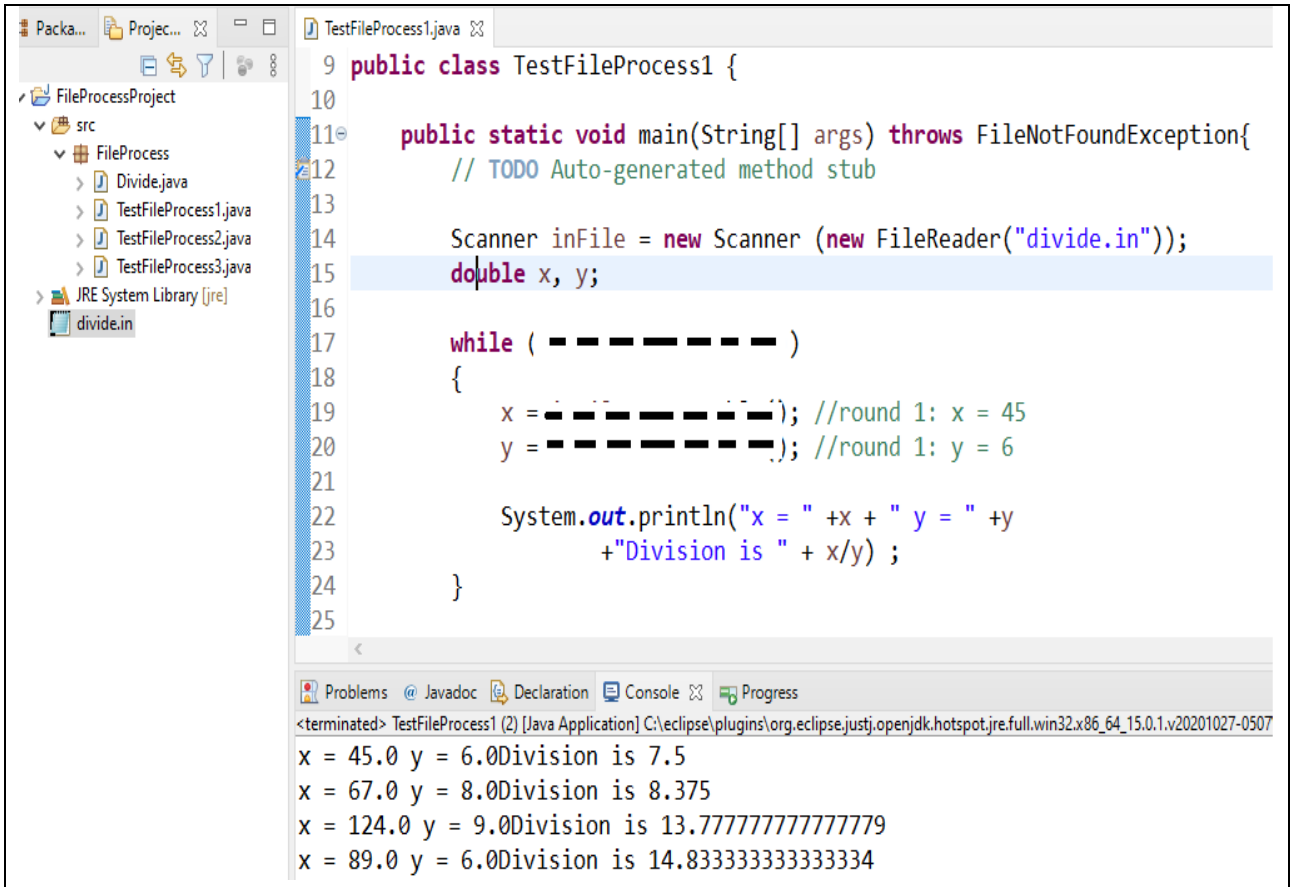


Figure 1

b) Reading Data From Text File and Storing it into Parallel Arrays

Create *TestFileProcess2.java* to read from input file and storing its content into parallel arrays as shown in Figure 2.

```java
public class TestFileProcess2 {
    public static void main(String[] args) throws FileNotFoundException{
        Scanner inFile = new Scanner (new FileReader("divide.in"));
        double[] [redacted]  //index x[0] x[1] x[2] x[3]
        double[] [redacted]
        int index=0;

        while ( [redacted])
        {
            x[index] = [redacted]); //round 1: x[0] = 45
            y[index] = [redacted]); //round 1: y[0] = 6

            index++;   //round 2: index = 1  x[1] = 67
        }
        inFile.close();//Like Closing db and free space memory

        //Print all array x and y. You can apply processing algorithms
        System.out.println("Printing Parallel Arrays");
        for (index=0; index< x.length; index++)
            System.out.println("x = " +x[index] + " y = " +y[index]
                    +" Division is " + x[index]/y[index]);
```

```
Printing Parallel Arrays
x = 45.0 y = 6.0 Division is 7.5
x = 67.0 y = 8.0 Division is 8.375
```

Figure 2

c) Reading Data From Text File and Storing it into Object of Arrays

Create *TestFileProcess3.java* to read from input file and storing its content into object of array as shown in Figure 3.

```java
public class TestFileProcess3 {

    public static void main(String[] args) throws FileNotFoundException{
        // TODO Auto-generated method stub

        Scanner inFile = new Scanner (new FileReader("divide.in"));
        Divide[] arrayInFile =[redacted];  //index arrayInfile[0] arrayInfile[1]

        int index=0;
        for (index=0; index< arrayInFile.length; index++)
            arrayInFile[ [redacted]

        index = 0;
        while (inFile.hasNextLine())
        {
            arrayInFile[index].set [redacted] //round 1: arrayInFile[0].
            arrayInFile[index].set [redacted] //round 1: arrayInFile[0].

            index++;   //round 2: index = 1  arrayInFile[1].x = 67
        }
        inFile.close();//Like Closing db and free space memory

        //Print all array x and y
        //you can apply processing algorithms

        System.out.println("Printing Array of Object");

        for (index=0; index< arrayInFile.length; index++)
            System.out.println("x = " +arrayInFile[index]. [redacted]
                    + " y = " +arrayInFile[index].g [redacted]
                    + " Division is " +
                    arrayInFile[index].getX()/arrayInFile[index].getY());
    }
}
```

```
Printing Array of Object
x = 45.0 y = 6.0 Division is 7.5
x = 67.0 y = 8.0 Division is 8.375
```

Figure 3

2. ArrayLists

a) Create *ArrayListProcessProject* using Eclipse IDE for reading input file *divide.in*. Create *TestArrayList1.java* to to create ArrayList from List interface as shown in Figure 4.

```
package ArrayListProcess;

import java.util.*;

public class TestArrayList1 {

    public static void main(String[] args) {

        List<String> namelist = new ArrayList
        String [ ] names = {"Ann", "Bob", "Carol"};

        int index=0;
        //Reading from Array names and filling up namelist arrayList
        for (index=0; index< names.length; index++)
            namelist.add(names[index]);

        //Print ArrayList namelist
        for (index=0; index< namelist.size(); index++)
            System.out.println(
    }

}
```

Figure 4

b) Reading Data from Text File and Storing it into ArrayList Data Structure
Create *TestFileArrayList2.java* to read from input file and storing its content into ArrayList as shown in Figure 5.

```
    public static void main(String[] args) throws FileNotFoundException{
        Scanner inFile = new Scanner (new FileReader("divide.in"));
        List inFileArrayList = new ArrayList();

        int index=0;
        //Reading from Input File names and filling up inFileArrayList
        while (inFile.hasNextLine())
        {
            Divide divideObj = new Divide();
            divideObj.setX(inFile.nextDouble());
            divideObj.setY(inFile.nextDouble());

            inFileArrayList.add
        }
        inFile.close();//Like Closing db and free space memory


        //Print ArrayList inFileArrayList
        for (index=0; index< inFileArrayList.size(); index++)
            System.out.println("x= "+inFileArrayList
                    +" y
                    .get(index).getY());
```

```
Problems  @ Javadoc  Declara
<terminated> TestFileArrayList2 [Java Ap
x= 45.0 y= 6.0
x= 67.0 y= 8.0
x= 124.0 y= 9.0
x= 89.0 y= 6.0
```

Figure 5

3. Using Array of Objects in GradeProject

a) Create a Java Project to be named ***GradeProject*** using Eclipse IDE to define a new data structure type, called *Grade*.

b) Create a class to define data structure type, called *Grade*, which is designed to group data and functions into a single unit that *represents* a template of the fields used in `Grade.in` as shown in the following Figure 6.

c) Each line within *Grade.in* represents a Grade's record with the following fields: `student_id (int), student_lname (string), student_fname (string), s_grade_Assignment1 (int), s_grade_Assignment2(int), s_grade_Assignment3(int), s_grade_Mid_Term (int), s_grade_Final_Term (int).`

d) Add **default constructor** `(student_id=0, student_fname="", student_lname="", s_grade_Assignment1=0,s_grade_Assignment2=0, s_grade_Assignment3=0,`

`s_grade_Mid_Term=0, s_grade_Final_Term=0)` and **constructor with parameters** (get the same name as the name of the class) within the *Grade* class in order to initialize the data members `(student_id,student_fname,student_lname,s_grade_Assignment1,s_grade_Assignment2, s_grade_Assignment3, s_grade_Mid_Term, s_grade_Final_Term)` of every object.

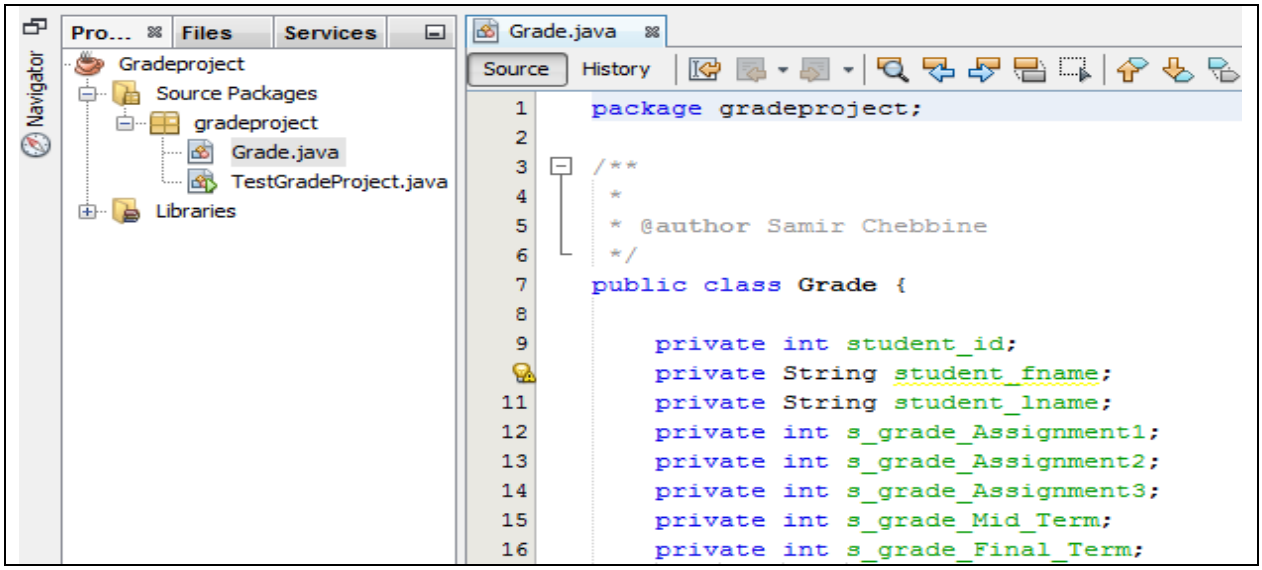
e) Add public **Mutator** (**setter**) methods in *Grade* class to modify the values of private members.

f) Add public **Accessor** (**getter**) methods in *Grade* class to read the values of private members.

g) Add a method called public String toString() in *Grade* class to print the *Publisher* information in the form of `"S Id :"+ student_id +"S LName :"+ student_lname + "S FName :" + student_fname + "S Ass1:" + s_grade_Assignment1 + "S Ass2:" + s_grade_Assignment2 + "S Ass3:" + s_grade_Assignment3 + "S MT:" + s_grade_Mid_Term + "S FT:" + s_grade_Final_Term.`

h) Add a method within *Grade* class called (`Calculate_GradeAverage ()`) that calculates and returns the average score for each student according to the following mark distribution:

1) 40 % for all assignments (*s_grade_Ass1* for Assignment 1), (*s_grade_Ass2* for Assignment 2), (*s_grade_Ass3* for Assignment 3)
2) 30 % for Mid Term Exam (*s_grade_Mt*)
3) 30 % for Final Exam (*s_grade_Ft*)



i) You need to *test* the implemented class `Grade` in the **main program** *TestGradeProject*.

j) Instantiate an array object of `Grade` class type to be referenced by (`all_sgrades`) of `Grade` class type) using the **default constructor**. Then read records from the input file `Grade.in` as shown in Figure 4 and assign the instance data of every *record* of the input File into its corresponding array object component using the implemented **setter** methods.

k) Invoke the method `Calculate_GradeAverage ()` related to every array object component, and display the student information using the **getter** methods, the grade average of every student, and the total grade average of all students as shown in the following Figure.

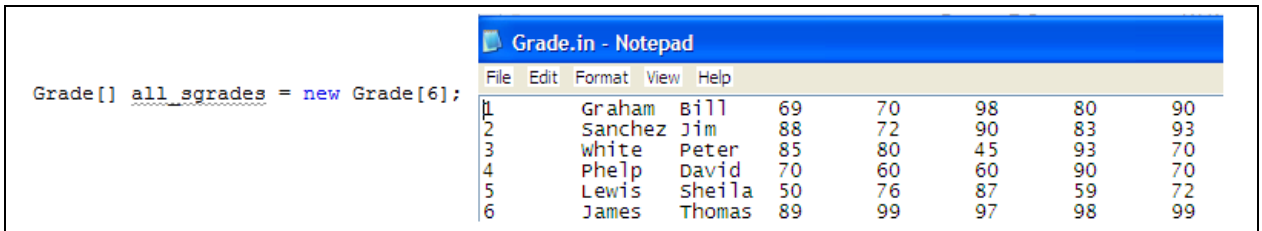


Figure 6

```
Output - Gradeproject (run)
run:

1) s_id: 0, Stdent Last Name: Graham, Student First Name: Bill

 Assignment 1 : 69
 Assignment 2 : 70
 Assignment 3 : 98
 Mid Term Exam : 80
 Final Exam : 90

 The Average score for this Student is 82.60


2) s_id: 2, Stdent Last Name: Sanchez, Student First Name: Jim

 Assignment 1 : 88
 Assignment 2 : 72
 Assignment 3 : 90
 Mid Term Exam : 83
 Final Exam : 93

 The Average score for this Student is 86.13


3) s_id: 3, Stdent Last Name: White, Student First Name: Peter

 Assignment 1 : 85
 Assignment 2 : 80
 Assignment 3 : 45
 Mid Term Exam : 93
 Final Exam : 70

 The Average score for this Student is 76.90
```

```
4) s_id: 4, Stdent Last Name: Phelp, Student First Name: David

 Assignment 1 : 70
 Assignment 2 : 60
 Assignment 3 : 60
 Mid Term Exam : 90
 Final Exam : 70

 The Average score for this Student is 73.33


5) s_id: 5, Stdent Last Name: Lewis, Student First Name: Sheila

 Assignment 1 : 50
 Assignment 2 : 76
 Assignment 3 : 87
 Mid Term Exam : 59
 Final Exam : 72

 The Average score for this Student is 67.70


6) s_id: 6, Stdent Last Name: James, Student First Name: Thomas

 Assignment 1 : 89
 Assignment 2 : 99
 Assignment 3 : 97
 Mid Term Exam : 98
 Final Exam : 99

 The Average score for this Student is 97.10

 Number of Students in the file is 6

 The Average score for all Students is 80.63

BUILD SUCCESSFUL (total time: 0 seconds)
```

l) (**Parallel Arrays**) Add more Java statements in the main program to do the following:

- Store the first column of the file *Grade.in* which represents the field `student_id` into *one-dimensional array* of *int* type, and print its components.
- Store the second and third columns of the file *Grade.in* which represents the fields `student_lname (string)`, `student_fname (string)` into *parallel one-dimensional array* of *String* type each, and print its components.
- Store all column grades `s_grade_Assignment1`, `s_grade_Assignment2`, `s_grade_Assignment3`, `s_grade_Mid_Term`, `s_grade_Final_Term` of the file *Grade.in* into *two dimensional array* of *double* type each, and print its components.

4. Using ArrayList Data Structure in ArrayListBillingProject

Create a Java Project **ArrayListBillingProject** using NetBeans IDE that allows end user to issue a billing by populating data file Billing.in into an **ArrayList** data structure of type Billing class type.

a) You need to design a **Java class** called **`Billing`**, which takes the price and quantity as two **private** `non static` members called respectively (`prd_Price`) and (`prd_Qty`). The variables called `Fed_Tax`, `Prv_Tax` as **public** and `static` data members.

1) Add **default constructor**, setters, getters, and toString()
2) Add a method called `CalculateBilling()` in `Billing` class to calculate the total of billing

```
T_Billing = (prd_Price* prd_Qty) + (prd_Price*prd_Qty)* Fed_Tax + (prd_Price*prd_Qty)* Prv_Tax
```

b) Create *TestArrayListBilling.java* where you populate an ArrayList data structure of Billing class type to be referenced by (billingArrList) from input file Billing.in. Set every component using the implemented setter methods (setPrd_Price(), setPrd_Qty()).

```
Billing.in
1  45.6 6 0.075 0.06
2  44.99 8 0.075 0.06
3  59.99 2 0.075 0.06

Reading from Billing.in Input file:
Printing ArrayList bilingArrList and perform processing

bilingArrList[0] Object: Billing [prd_Price=45.6, prd_Qty=6]
        The Total of Billing of bilingArrList[0] object is 310.54$

bilingArrList[1] Object: Billing [prd_Price=44.99, prd_Qty=8]
        The Total of Billing of bilingArrList[1] object is 408.51$

bilingArrList[2] Object: Billing [prd_Price=59.99, prd_Qty=2]
        The Total of Billing of bilingArrList[2] object is 136.18$

All Total of Billing is 855.22$
Thank you for Doing Business With Us
```

Calculate the total of billing of all ArrayList components. Display the total as shown above.