INFORMATICS INSTITUTE OF TECHNOLOGY

Software Development II

Coursework Report 2023/2024

Warsha Diwyanjali

w2084393

20230179

Task 01 – Source Code

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;
import java.util.jar.Attributes;

public class studentActivityManagementSystem {
   private static final int maxSize = 100;
   private static Student[] studentsArr = new Student[maxSize];
   private static int studentCount = 0;

   public static void checkAvailableSeats(){
      System.out.println("Available Seats: "+(maxSize-studentCount));
   }


   public static void registerStudent(Scanner scanner){
      String id="";
      double modMark=0;
      boolean flag=false;
      while(true){
         flag=false;
         System.out.print("Enter student id: ");
         id=scanner.next();
         if(id.length()==8){
            for(int i=0;i<studentsArr.length;i++){
               if(studentsArr[i]!=null){
```

```
                Student student=studentsArr[i];
                if(student.getStuId().equals(id)){
                    System.out.println("That student is already exists.");
                    flag=true;
                    break;
                }
            }
        }
        if(!flag){
            break;
        }
    }else{
        System.out.println("Enter a id with length of 8.");
    }
}
scanner.nextLine();
System.out.print("Enter student name: ");
String name=scanner.nextLine();
String listOfNames[]=name.split(" ");
for(int i=0;i<listOfNames.length;i++){ // Capitalize the first letter of the eaxh word in the name
    listOfNames[i]=listOfNames[i].substring(0,1).toUpperCase()+listOfNames[i].substring(1);
}
String first=listOfNames.length>0?listOfNames[0].trim():"";
String second=listOfNames.length>1?listOfNames[1].trim():"";

name=((listOfNames.length>1)?first+" ":first+"")+second;//If student entered the name of two
words arranging space between them

Module moduleArr[]=new Module[3];
```

```
        for(int i=0;i<moduleArr.length;i++){
            Module module=new Module(modMark);
            moduleArr[i]=module;
        }
        Student student=new Student(name,id,moduleArr);
        studentsArr[studentCount++]=student; //Adding the student to the array
        System.out.println("Student registered Successfully.");
    }

    public static void deleteStudent(Scanner scanner){
        int count=0;
        Student temp[]=new Student[maxSize]; //Created a temorary array to copy the new students without the deleted student
        boolean flag=false;
        if(studentsArr[0]==null){
            System.out.println("No students registered yet.");
        }
        while(true){
            System.out.print("Enter student id: ");
            String id=scanner.next();
            if(id.length()==8){
                for(int i=0;i<studentsArr.length;i++){
                    if(studentsArr[i]!=null){
                        Student student=studentsArr[i];
                        if(student.getStuId().equals(id)){
                            flag=true;
                            count=i;
                            break;
                        }
```

```
            }
          }
            break;//Breaking the while loop after checking entered student in the array or not in the array
        }else{
            System.out.println("Enter a id with 8 characters.");
        }
      }
      for(int i=0;i<count;i++){ //Copying the elements selected id
        temp[i]=studentsArr[i];
      }
      for(int j=count+1;j<temp.length;j++){ //Copying the elements after the equals student id in to a temporary array
        temp[j]=studentsArr[j];
      }
      studentsArr=temp; //Sharing the reference of the temporary array
      if(!flag){
        System.out.println("No such student");
        return;
      }
      System.out.println("Student deleted successfully.");
    }

    public static void storeDetailsToFile(){
      try {
        FileWriter detailsFile=new FileWriter("text.txt");
        for(int i=0;i<studentsArr.length;i++){
          if(studentsArr[i]!=null){ //checking student array i place is an empty
            Student student=studentsArr[i];
            Module[]modules=student.getModules();
```

```
            detailsFile.write(student.getStuId()+","+student.getStuName()+",");// Write student name
and id
            for(int l=0;l<3;l++){
              if(modules[l]!=null){ //Created a if statement for check student have module marks
                detailsFile.write(modules[l].getModuleMark()+(l==2?"":","));    //Removed additional
comma
              }
            }
            detailsFile.write("\n");
          }
        }
        detailsFile.close();
        System.out.println("Student details successfully stored.");
      } catch (IOException e) {
        System.out.println("Text file not found.");
      }
    }
    public static void loadStudentDetailsFromFile(){
      int counter=0; //Declare and initialize for better use
      String name="";
      try {
        File file =new File("text.txt");
        Scanner reader=new Scanner(file);
        while(reader.hasNextLine()){ //Looping while text file last line
          String text=reader.nextLine(); //Reading line by line
          String[] listOfDetails=text.split(","); //remove commas and seperate
          String id=listOfDetails[0].trim();
          name=listOfDetails[1];
          Module modules[]=new Module[3];
```

```
            for(int num=0;num<modules.length;num++){


                double modMark=Double.parseDouble(listOfDetails[2+num].trim());//Use to convert a string
in to a double
                modules[num]=new Module(modMark);


            }
            Student student=new Student(name,id,modules); //Creating student object and entering to the
array
            studentsArr[counter++]=student;
          }
          studentCount=counter;
          reader.close();
          System.out.println("Successfully lorded students to the array.");


       }catch (Exception e) {
          System.out.println("File not found");
       }
    }



    public static void findStudent(Scanner scanner){
       String id;
       int studentObj=0;
       boolean flag=false;
       if(studentsArr[0]==null){ //If studentsArr first element equals to null
          System.out.println("No students registered.");
          return;
       }
```

```
while(true){ //while student id length equals to 8
   System.out.print("Enter your ID: ");
   id=scanner.next();
   if(id.length()==8){
      for(int i=0;i<studentsArr.length;i++){
         if(studentsArr[i]!=null){
            Student student=studentsArr[i];
            if(student.getStuId().equals(id)){// if entered it equals to student id
               flag=true;
               studentObj=i;
               break;
            }
         }
      }
      if(flag){
         Student student=studentsArr[studentObj];
         System.out.println("Student Name: "+student.getStuName());
         Module modules[]=student.getModules();
         for(int y=0;y<modules.length;y++){ //Printing 3 module marks of that student
            System.out.println("Module "+(y+1)+" Mark: "+modules[y].getModuleMark());
         }
         break;
      }else{
         System.out.println("No Such Student.");
         return;
      }
   }else{
      System.out.println("Enter a id with 8 characters.");
   }
```

```
    }
}

public static void viewTheStudentWithNames(){
    //Entering all the student names in to the array
    String NamesArr[]=new String[studentCount];
    for(int i=0;i<studentsArr.length;i++){
        if(studentsArr[i]!=null){
            Student student=studentsArr[i];
            NamesArr[i]=student.getStuName();
        }
    }
    //Sorting in alphabetical order
    for(int i=0;i<NamesArr.length;i++){
        for(int y=i;y<NamesArr.length;y++){
            if(NamesArr[i].compareTo(NamesArr[y])>0){ //Compare unicode values
                String temp=NamesArr[i];
                NamesArr[i]=NamesArr[y];
                NamesArr[y]=temp;

            }
        }
    }
    //Printing sorted names
    System.out.print("Student names in alphabat order: ");
    for(int i=0;i<NamesArr.length;i++){
        System.out.print(NamesArr[i]+", ");
    }
    System.out.println("\b\b "); //Remove additional comma
```

```
}

public static void subMenu(){
   System.out.println("Sub Menu");
   System.out.println("\t1. Add Student Marks");
   System.out.println("\t2. Summary");
   System.out.println("\t3. Report");
   System.out.println("\t4. Return to main menu");
}

public static void additionalOptions(Scanner scanner){
   int choice=0;
   boolean flag=false;
   while(true){
      try {
         subMenu();
         System.out.print("Enter your choice: ");
         choice=scanner.nextInt();
      } catch (Exception e) {
         System.out.print("Enter a valid integer value.");
         scanner.nextInt();// Clear the input
      }
      switch(choice){ //Checking the option using a switch
         case 1:addStudentMarks(scanner); break;
         case 2:summary(); break;
         case 3:report(); break;
         case 4:flag=true; break;
         default:System.out.println("Enter a integer in the options");
      }
```

```
            if(flag){
                System.out.println("Exiting Sub Menu...");
                break;
            }
        }
    }

    public static void addStudentMarks(Scanner scanner){
        boolean flag=false;
        if(studentsArr[0]==null){
            System.out.println("No students are added yet.");
            return;
        }
        scanner.nextLine();
        System.out.print("Enter student name: ");
        String name=scanner.nextLine();
        String namesArr[]=name.split(" ");// removes " " and sperate
        for(int i=0;i<namesArr.length;i++){
            namesArr[i]=namesArr[i].substring(0,1).toUpperCase()+namesArr[i].substring(1);//capitalize
every letter in each word
        }
        String first=namesArr.length>0?namesArr[0]:"";
        String second=namesArr.length>1?namesArr[1]:"";//If student has 2 words in the name, space
handling

        name=namesArr.length<2?first+"":first+" "  +(second.isEmpty()?"":second.trim());

        for(int i=0;i<studentsArr.length;i++){
            if(studentsArr[i]!=null){
```

```
            Student student=studentsArr[i];
            if(student.getStuName().equals(name)){ //Check entered name and student name is equals
               flag=true;
               Module moduleArr[]=student.getModules();
               for(int s=0;s<moduleArr.length;s++){
                  System.out.print("Enter module "+(s+1)+" mark: ");
                  double moduleMark=scanner.nextDouble();
                  Module module=new Module(moduleMark);
                  moduleArr[s]=module;
               }
               System.out.println("Student marks added successfully.");
               break;

            }
         }
      }
      if(!flag){
         System.out.println("No Such Student.");
      }

   }
   public static void main(String args[]){
      int choice=0;
      while(true){
         displayMenu();
         boolean flag=false;
         Scanner scanner=new Scanner(System.in); //Take the Scanner class as the scanner
         while(true){
            try{
```

```
            System.out.print("Enter your choice: ");
            choice=scanner.nextInt();
            break;
        }catch(Exception e){
            System.out.println("Invalid number");
        }
    }
    switch(choice){
        case 1:checkAvailableSeats();break;
        case 2:registerStudent(scanner); break;
        case 3:deleteStudent(scanner); break;
        case 4:findStudent(scanner); break;
        case 5:storeDetailsToFile(); break;
        case 6:loadStudentDetailsFromFile(); break;
        case 7:viewTheStudentWithNames(); break;
        case 8:additionalOptions(scanner); break;
        case 9:flag=true; break;
        default:System.out.println("Invalid input.");
    }
    if(flag){
        System.out.println("Exiting...");
        break;
    }
  }
}

private static void displayMenu() {
  System.out.println("\nMenu :");
  System.out.println("1. Check available seats");
```

```
        System.out.println("2. Register student");
        System.out.println("3. Delete student");
        System.out.println("4. Find student");
        System.out.println("5. Store student details into a file"); // Serialization is not supported
        System.out.println("6. Load student details from the file to the system"); // Deserialization is not supported
        System.out.println("7. View the list of students based on their names");
        System.out.println("8. Additional Options");
        System.out.println("9. Exit");
    }



}
```

Task 02 – Source Code

```
public class Student {
    private String studentId;
    private String studentName;
    private Module modules[];
    public Student(String studentName,String studentId,Module modules[]){
        this.studentId=studentId;
        this.studentName=studentName;
        this.modules=modules;
    }
    public String getStuName(){
        return studentName;
    }
    public String getStuId(){
        return studentId;
    }
    public Module[] getModules(){
        return modules;
    }
}
```

```
public class Module {
    private double moduleMark;
    public Module(double moduleMark){
        this.moduleMark=moduleMark;
    }

    public double getModuleMark(){
        return moduleMark;
    }
}
```

Task 03 – Source Code

```
public static void summary(){
    boolean flag=false;
    int counter=0;
    System.out.println("Total Students Registered: "+studentCount);

    for(int i=0;i<studentsArr.length;i++){
        if(studentsArr[i]!=null){
            Student student=studentsArr[i];
            Module modArr[]=student.getModules();
            for(int p=0;p<modArr.length;p++){
                if(modArr[i].getModuleMark()>40){
                    flag=true;// flag for handle the all the module marks greater than 40
                }else{
                    flag=false;
                    break;
                }
            }
            if(flag){
                counter++;
            }
        }
    }
    System.out.println("Total students get more than 40 marks for all 3 modules: "+counter);
}
```

```
public static void report(){
    double total=0;
    double average=0;
    String grade="";
    int count=0;
    for(int i=0;i<studentsArr.length;i++){
        total=0;
        if(studentsArr[i]!=null){ //checking the array element is null or not
            Student student=studentsArr[i]; //Taking object to the variable
            Module moduleArr[]=student.getModules();
            System.out.println("Student ID: "+student.getStuId());
            System.out.println("Student Name: "+student.getStuName());
            for(Module module:moduleArr){
                count++;
                System.out.println("Module "+(count+1)+" Mark: "+module.getModuleMark());
                total+=module.getModuleMark();
            }
            average=total/3;
            System.out.println("Total is : "+total);
            System.out.println("Average is : "+average);
            if(average>=80){
                grade="Distinction";
            }else if(average>=70){
                grade="Merit";
            }else if(average>=40){
                grade="Pass";
            }else{
                grade="Fail";
            }
```

```
            System.out.println("Grade: "+grade);
            System.out.println();
        }
    }

System.out.println("**********************************************************************
****");
  }
```

Task 04 – Testing

Test Case	Expected Result	Actual Result	Pass/Fail
Check available seats	Enter 1 to check available seats	Enter 1 to check available seats	pass
Register a student	Enter 2 register a student by their student id and name.	Enter 2 register a student by their student id and name.	pass
Delete a student	Enter 3 to delete a student by their id	Enter 3 to delete a student by their id	pass
Find a student	Enter 4 to find a student by their id	Enter 4 to find a student by their id	pass
Store student details into a text file	Enter 5 store students details into a file	Enter 5 store students details into a file	pass
Load student details from the file to the system	Enter 6 to load the data back to the system again	Enter 6 to load the data back to the system again	Pass
View the list of students based on their names	Enter 7 to view the list of student's names in alphabetical order.	Enter 7 to view the list of student's names in alphabetical order.	Pass
Additional Options	Enter 8 to select additional options	Enter 8 to select additional options	pass
	Enter 1 to add marks for module 1,2 and 3	Enter 1 to add marks for module 1,2 and 3	pass
	Enter 2 to get a summary of total student registrations, total no of students who are scored more than 40 marks in Module 1, 2 and 3.	Enter 2 to get a summary of total student registrations, total no of students who are scored more than 40 marks in Module 1, 2 and 3.	pass

	Enter 3 to generate complete report with list of students includes student ID, student name, module 1 marks, module 2 marks, module 3 marks, total, average and grade	Enter 3 to generate complete report with list of students includes student ID, student name, module 1 marks, module 2 marks, module 3 marks, total, average and grade	pass
	Enter 4 to return to the main menu	Enter 4 to return to the main menu	pass
Exit	Enter 9 to finish process with exit code	Enter 9 to finish process with exit code	pass

Task 04 – Testing - Discussion

These test cases have chosen to ensure that test cases cover all parts of StudentActivityManagementSystem. While ensuring that the system runs smoothly from registration to data persistence and retrieval, and from data management to report generation, the tests intend to meticulously validate the system's toughness and dependability by covering different situations including boundary conditions (e.g. ID length, unique ID), storing and loading, and data management.

Arrays and Classes

An array is a group of variables of the same data type and referred to by a common name. An array is a block of consecutive memory locations that hold values of the same data type.

A class presents a template or a blueprint of an object, its data and its methods. Structure of a class contains Data and methods.

The array-based solution is direct forward and easy to deploy, hence it is suitable for small datasets or applications with fixed number of elements. For instance, accessing directly to the elements by their indices is highly efficient in certain cases when using an array. But there are several important drawbacks associated with this approach, such as having a maximum size that does not change ever, which may lead to "wasted memory" or "large student bodies" which cannot be processed at all. Furthermore, maintaining and extending such a solution based on arrays is difficult because new functionality may lead to major modifications being performed at every place where the code is used.

On the other side, class-based solutions benefit from all the power brought in by object-oriented programming: enhanced data encapsulation, reuse along with the scalability. This approach is excellent for applications that are going to change or be updated frequently. It facilitates encapsulation mechanisms for handling and reuse of data and for the extension of classes up to a point with a view to making a system easier to maintain and extend. The class-based approach is also constructed with an eye to variation in the number of students, which also enhances the stability of the solution. This could bring in additional discomfort in enhancing the small, uncomplicated applications and might bring in slightly more overhead from dealing with objects and invoking methods.

Considering all the above-mentioned advantages and disadvantages, the class-based approach, indeed, has immense benefits in the aspects of maintainability, scalability, and readability to make the development of StudentActivityManagementSystem most preferable.

Self-Evaluation form

Criteria	Allocated marks	Expected marks	Total
Task 1 Three marks for each option (1,2,3,4,5,6,7,8)	24	24	**(30)**
Menu works correctly	6	6	
Task 2 Student class works correctly	14	14	**(30)**
Module class works correctly	10	10	
Sub menu (A and B works well)	6	5	
Task 3 Report – Generate a summary	7	7	**(20)**
Report – Generate the complete report	10	10	
Implementation of Bubble sort	3	3	
Task 4 Test case coverage and reasons	6	4	**(10)**
Writeup on which version is better and why.	4	4	
Coding Style (Comments, indentation, style)	7	5	**(10)**
Complete the self-evaluation form indicating what you have accomplished to ensure appropriate feedback.	3	3	
Totals	100	95	**(100)**

References

W3Schools (2019). *Java Arrays*. [online] W3schools.com. Available at: https://www.w3schools.com/java/java_arrays.asp.

www.javatpoint.com. (n.d.). *Bubble Sort in Java - Javatpoint*. [online] Available at: https://www.javatpoint.com/bubble-sort-in-java.

W3Schools (2019). *Java Classes and Objects*. [online] W3schools.com. Available at: https://www.w3schools.com/java/java_classes.asp.

edureka (2019). *File Handling in Java | Reading and Writing File in Java | Java Training | Edureka*. *YouTube*. Available at: https://www.youtube.com/watch?v=SslMi6ptwH8.