Experiment No. 2

Title:Implement the concept of array of an object

Batch:SY-IT(B3) **Roll No.:16010423076** **Experiment No.:2**

Aim: Write a program to create an array of the objects of class student. The class should have field members name, id, total marks and marks in three subjects namely JAVA, DSA, DBMS. Accept information of 'n' students and display them in tabular form in descending order of total marks obtained by student

---

Resources needed: java

Theory:

An array of objects is created just like an array of primitive type data items in the following way.

Student[] studentArray = new Student[7];

The above statement creates the array which can hold references to seven Student objects. It doesn't create the Student objects themselves. They have to be created separately using the constructor of the Student class. The studentArray contains seven memory spaces in which the address of seven Student objects may be stored. If we try to access the Student objects even before creating them, run time errors would occur. For instance, the following statement throws a NullPointerException during runtime which indicates that studentArray[0] isn't yet pointing to a Student object.

studentArray[0].marks =100;

The Student objects have to be instantiated using the constructor of the Student class and their references should be assigned to the array elements in the following way.

studentArray[0] = new Student();

In this way, we create the other Student objects also. If each of the Student objects have to be created using a different constructor, we use a statement similar to the above several times. However, in this particular case, we may use a for loop since all Student objects are created with the same default constructor.

```
for ( int i=0; i<studentArray.length; i++) {

studentArray[i]=new Student();

}
```

The above for loop creates seven Student objects and assigns their reference to the array elements. Now, a statement like the following would be valid.

studentArray[0].marks=100;

Enhanced for loops find a better application here as we not only get the Student object but also we are capable of modifying it. This is because of the fact that Student is a reference type.

Therefore the variable in the header of the enhanced for loop would be storing a refer - ence to the Student object and not a copy of the Student object which was the case when primitive type variables like int were used as array elements.

```
for ( Student x : studentArray ) {

x.marks = s.nextInt(); // s is a Scanner object

}
```

Results: (Program with output)

```
import java.util.Scanner;
class Student
{
 String name;
 int rollno;
 int javamarks, dsamarks, dbmsmarks, totalmarks;

 void input(Scanner sc)
 {
 System.out.println("Enter the name of student: ");
 name = sc.nextLine();
 System.out.println("enter the roll no. of student: ");
 rollno = sc.nextInt();
 System.out.println("Enter the marks in Java: ");
 javamarks = sc.nextInt();
 System.out.println("Enter the DSA marks: ");
 dsamarks = sc.nextInt();
 System.out.println("Enter the DBMS marks: ");
 dbmsmarks = sc.nextInt();
 totalmarks = javamarks + dsamarks + dbmsmarks;
 sc.nextLine();
 }

 void display()
 {
 System.out.printf("%-20s %-10d %-10d %-10d %-10d %-10d\n", name, rollno, javamarks,
dsamarks, dbmsmarks, totalmarks);
System.out.println();
  }

 public static void main(String[] args)
 {
 Scanner sc = new Scanner(System.in);
 System.out.println("Enter no. of students: ");
 int numofstud = sc.nextInt();
 sc.nextLine();

 Student[] studarr = new Student[numofstud];
for (int i = 0; i<numofstud; i++)
 {
 studarr[i] = new Student();
 studarr[i].input(sc);
 }

 // bubble sort
 for (int i = 0; i<numofstud-1; i++)
 {
 for (int j = 0; j<numofstud-i-1; j++)
 {
 if (studarr[j].totalmarks < studarr[j+1].totalmarks)   {
 Student temp = studarr[j];
 studarr[j] = studarr[j+1];
 studarr[j+1] = temp;
 }
 }
 }

 for (int i = 0; i<numofstud; i++)
 {
 studarr[i].display();
 }
 sc.close();
 }
}
```

Output :

```
Enter no. of students:
2
Enter the name of student:
Ritesh
enter the roll no. of student:
76
Enter the marks in Java:
99
Enter the DSA marks:
98
Enter the DBMS marks:
99
Enter the name of student:
Rohit
enter the roll no. of student:
25
Enter the marks in Java:
77
```

```
Enter the DSA marks:
89
Enter the DBMS marks:
99

Ritesh 76 99 98 99 296

Rohit 25 77 89 99 265


=== Code Execution Successful ===
```

Questions:

Explain Jagged array.

A jagged array is one whose elements are an array. All these arrays of different lengths are unlike multidimensional arrays where each sub-array has the same length.

Properties of Jagged Arrays:

Non-uniform Length - The length of individual sub-arrays could be different.

Memory Efficiency - It is a way to save more memory on structures that do not require uniform rows and columns.

Flexibility: This is useful in those situations when data naturally forms a non-rectangular structure.

Syntax :

```
int[][] jaggedArray = new int[3][]; // Declare a jagged array with 3 rows
// Initialize each row with different lengths
jaggedArray[0] = new int[2]; // First row has 2 elements
jaggedArray[1] = new int[4]; // Second row has 4 elements
jaggedArray[2] = new int[3]; // Third row has 3 elements
```

Benefits of Using Jagged Arrays:

1. **Flexibility**: Allows for more flexible data structures when the rows of the array have different sizes.
2. **Memory Efficiency**: Saves memory when dealing with data that doesn't naturally fit into a rectangular shape.

When to Use Jagged Arrays:

Jagged arrays are useful when dealing with situations where:

• Each row or sub-array can have a different number of elements.

• You need to represent data that is inherently non-uniform, like adjacency lists in graphs, or marks of students in different numbers of subjects.

Outcomes:

CO1: Apply fundamental Object Oriented Methodology concepts using java programming CO2: Apply String manipulation functions ,inheritance and polymorphism using Java programming

Conclusion: (Conclusion to be based on the outcomes achieved)

From this article I learnt creating a student array in Java. This Java program creates a Student class to input and display students' details, the marks obtained in three subjects and the total marks. Then it sorts these students in descending order of total marks using a bubble sort algorithm and displays them. The approach is given to understand the primary concepts of programming: object-oriented design, array handling, and basic sorting algorithms, which makes it practical in understanding data management in Java.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

1. Herbert Schildt, “Java: The Complete Reference” Tata McGrawHill Publishing Company Limited Tenth Edition, 2017
2. Sachin Malhotra, Saurab Choudhary, “Programming in Java” Oxford University Press Second Edition, 2018 3.
3. D.T. Editorial Services, “Java 8 Programming Black Book” Dream tech Press Edition 2015.