SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE CODE: DJS22ITL306 **DATE:2/12/23**

COURSE NAME: Programing Laboratory 1 (Advanced Java) **DIVISION:I1-1**

CLASS: S.Y B. Tech IT **SAPID:60003220045**

NAME: ANISH SHARMA

EXPERIMENT NO. 6

CO/LO:
CO1- Modify the behaviour of methods, classes, and interfaces at runtime.

AIM / OBJECTIVE:

Use reflection API to examine or modify the behavior of methods, classes, and interfaces at runtime.

PROBLEM STATEMENTS:

Create a class student with private members attendance and marks. Create a class teacher who sets the values for marks and attendance. Finally create a class parent who creates a reflection of methods to know the values of marks and attendance of the student.

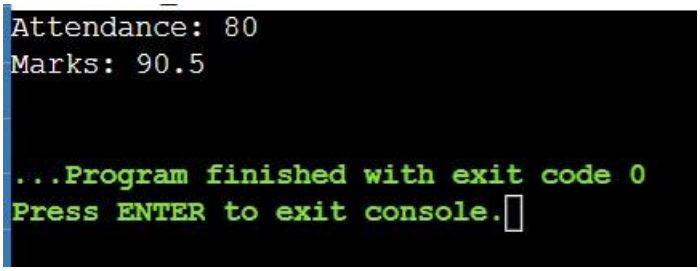
PROGRAM:

```
import java.lang.reflect.Field; import
java.lang.reflect.Method; import
java.lang.reflect.Constructor; class
Student {     private int attendance;
   private double marks;

   public int getAttendance() {
      return attendance;
   }

   public void setAttendance(int attendance) {
this.attendance = attendance;
   }

   public double getMarks() {
```

```
        return marks;
    }

    public void setMarks(double marks) {
                                                                this.marks = marks;
    }
}

class Teacher {
    public static void setValues(Student student, int attendance, double marks) throws Exception {
        Class studentClass = student.getClass();
//student obj is passed cuz we need the same obj to modify the attendence and marks
//field is used to access variables
        Field attendanceField = studentClass.getDeclaredField("attendance");
attendanceField.setAccessible(true);
        attendanceField.set(student, attendance);

        Field marksField = studentClass.getDeclaredField("marks");
marksField.setAccessible(true);
        marksField.set(student, marks);
    }
}

class Parent {
    public static void main(String[] args) throws Exception {
        Student student = new Student();
        Teacher.setValues(student, 80, 90.5);
        Class studentClass = student.getClass();
        Method getAttendanceMethod = studentClass.getDeclaredMethod("getAttendance");
        int attendance = (int) getAttendanceMethod.invoke(student);
        System.out.println("Attendance: " + attendance);
        Method getMarksMethod = studentClass.getDeclaredMethod("getMarks");
        double marks = (double) getMarksMethod.invoke(student);
        System.out.println("Marks: " + marks);
    }
}
```

SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

OUTPUT:



CONCLUSION:

This Java code demonstrates the use of reflection to dynamically access and modify private fields of a `Student` object through a separate `Teacher` class. The `Parent` class then utilizes reflection to invoke the `getAttendance` and `getMarks` methods, retrieving and printing the values of attendance and marks, respectively. This approach allows indirect access to private members, emphasizing the flexibility and power of reflection for manipulating class internals.