Java School Management System Simulator using Object Oriented Programming Language

The School Management System simulator examples using most Covers in Object Oriented Programing in features etc., Class, object, inheritance, polymorphism, Encapsulation...

Student Blueprint and Templates

1.State/Attribute

2.Student Name

3.Student id

4.Student grade

5.Behavior /Methods

6.Student (String Name, int id, int grade)

7.get id ()

8.get name ()

9.get grade ()

10.get Fees paid ()

11.get Fees Total ()

12.get Remaining Fees ()

Answer

```
/**
  * This class is responsible for keeping the
  * Track of student's fees, name, grade & fees
  * Paid.
  */
```

```
public class Student {
      private int id;
       private String name;
         private int grade;
          private int fees Paid;
         private int fees Total;
    /**
     * To create a new student by initializing.
     * Fees for every student is $20,000.
     * Fees paid initially is 0.
     * @param id for the student: unique.
     * @param name of the student.
     * @param grade of the student.
     * @return
     */
    public Student (int id, String name, int grade) {
          this. Fees Paid=0;
         this. fees Total=20000;
        this.id=id;
        this.name=name;
        this. Grade =grade;
    }

    //Not going to alter student's name, student's id.
    /**
```

```
   *
   * @return the id of the student
   */
  public int get Id () {
      return id;
  }
/**
   *
   * @return name of the student.
   */
  public String get Name () {
      return name;
  }
/**
   *
   * @return the grade of the student.
   */
  public int get Grade () {
      return grade;
  }
 /**
   *
   * @return fees paid by the student.
   */
  public int get Fees Paid () {
        return fees Paid;
    }
/**
     *
     * @return the total fees of the student.
     */
    public int get Fees Total () {
        return fees Total;
    }
 /**
     *
     * @return the remaining fees.
     */
    public int get Remaining Fees () {
        return fees Total-fees Paid;
    }
    /**
     * Used to update the student's grade.
     * @param grade new grade of the student.
     */
    public void set Grade (int grade) {
        this. grade =grade;
    }
    /**
     * Keep adding the fees to fees Paid Field.
     * Add the fees to the fees paid.
     * The school is going receive the funds.
     *
     * @param fees the fees that the student pays.
     */
    public void pays Fees (int fees) {
        fees Paid+=fees;
        School.updateTotalMoneyEarned(fees Paid);
    }
@Override
    public String to String () {
        return "Student's name:"+name+
                " Total fees paid so far $"+ fees Paid;
    }
}
```

Teacher Blueprint and Templates

1.State/Attribute

2.Teacher id

3. Teacher Name

4.Teacher Salary

5. Teacher Salary Earned

5.Behavior /Methods

6.Teacher (int id, String Name, int Salary, int Salary Earned=0,)

7.get id ()

8.get name ()

9.set Salary ()

Answer

```
/**
 * This class is responsible for keeping the track
 * Of teacher's name, id, salary.
 */
public class Teacher {
   private int id;
   private String name;
   private int salary;
   private int salary Earned;
    /**
    * Creates a new Teacher object.
    * @param id for the teacher.
    * @param name of the teacher.
    * @param salary of the teacher.
    */
   public Teacher (int id, String name, int salary) {
      this.id=id;
      this.name=name;
      this. salary =salary;
      this. salary Earned=0;
   }
/**
    *
```

```
     * @return the id of the teacher.
     */
    public int get Id () {
        return id;
    }
/**
     *
     * @return name of the teacher.
     */
    public String get Name () {
        return name;
    }
 /**
     *
     * @return the salary of the teacher
      */
    public int Salary () {
         return salary;
    }
    /**
     * Set the salary.
     * @param salary
     */
    public void set Salary (int salary) {
        this. salary =salary;
    }
    /**
        * Adds to salary Earned.
        * Removes from the total money earned by the school.
        * @param salary
     */
    public void receives Salary (int salary) {
        salary Earned+=salary;
        School. Update Total Money Spent (salary);
    }
 @Override
    public String to String () {
        return "Name of the Teacher: " + name
                +" Total salary earned so far $"
                + salary Earned;
    }
}
```

School Blueprint and Templates

1.State/Attribute

2.Student Name

3.Student id

5.Behavior /Methods

6.Student (String Name, int id, int grade)

7.get id ()

8.get name ()

9.get grade ()

10.get Fees paid ()

11.get Fees Total ()

12.get Remaining Fees ()

Answer

```
import java. util. List;
/**
   * Many teachers, many students.
   * Implements teachers and student
   * Using an Array List.
 */
public class School {
        private List<Teacher>teachers;
        private List<Student> students;
    private static int total Money Earned;
    private static int total Money Spent;
  /**
     * New school object is created.
     * @param teachers list of teachers in the school.
     * @param students list of students in the school.
     */
    public School (List<Teacher> teachers, List<Student> students) {
        this. teachers = teachers;
        this. students = students;
        total Money Earned=0;
        total Money Spent=0;
    }
 /**
     *
     * @return the list of teachers in the school.
     */
    public List<Teacher> get Teachers () {
        return teachers;
    }
/**
     * Adds a teacher to the school.
     * @param teacher the teacher to be added.
     */
    public void Add Teacher (Teacher teacher) {
        teachers. add (teacher);
    }
 /**
     *
     * @return the list of students in the school.
     */
    public List<Student> get Students () {
        return students;
    }
/**
     * Add a student to the school
     * @param student the student to be added.
     */
    public void add Student (Student student) {
        students. add (student);
    }
/**
     *
     * @return the total money earned by the school.
     */
    public int get Total Money Earned () {
        return total Money Earned;
    }
/**
     * Adds the total money earned by the school.
     * @param Money Earned money that is supposed to be added.
     */
    public static void update Total Money Earned (int Money Earned) {
        total Money Earned += Money Earned;
    }
/**
     *
     * @return the total money spent by the school.
     */
    public int get Total Money Spent () {
        return total Money Spent;
    }
 /**
     * Update the money that is spent by the school which
     * Is the salary given by the school to its teachers.
     * @param money Spent the money spent by school.
     */
   public static void update Total Money Spent (int money Spent) {
       total Money Earned-=money Spent;
    }
}
```

Main Blueprint and Templates

```
import java. util. Array List;
import java. util. List;
public class Main {
public static void main (String [] args) {
        Teacher anusha = new Teacher(1,"anusha",500);
        Teacher divya = new Teacher(2,"divya",700);
        Teacher Sirisha = new Teacher(3,"sirisha",600);
        List<Teacher> teacher List = new Array List<> ();
        teacherList.add(anusha);
        teacherList.add(divya);
        teacher List. Add (Sirisha);
        Student Teja = new Student(1,"Teja",4);
        Student Geetha = new Student (2,"Geetha", 012);
        Student Mounika = new Student (3,"Mounika", 015);
        List<Student> student List = new Array List<> ();
        Student List. add (Teja);
        Student List. add (Geetha);
        student List. Add (Mounika);
        School college1 = new School (teacher List, student List);
        Teacher teacher4 = new Teacher (6,"teacher4”,900);
        college1.addTeacher(teacher4);
        Teja. pay Fees (5000);
        geetha. pay Fees (6000);
        System. Out. Println ("college1 has earned $”
+college1.getTotalMoneyEarned());
    System. out. Println ("------Making SCHOOL PAY SALARY----");
        anusha. receive Salary (anusha. Salary ());
System.out. Println ("college1 has spent for salary to " + anusha. Get
Name ()
        +" and now has $" + college1.getTotalMoneyEarned());
       divya. Receive Salary (divya. Salary ());
        System. out. Println ("college1 has spent for salary to " + divya.
get Name ()
                +" and now has $" + college1.getTotalMoneyEarned());
         System. out. Println (geetha);
        Sirisha. Receive Salary (Sirisha. Salary ());
         System. out. println (Sirisha);
             }
      }
```