

```

// Case Study 9.1: TestScoresModel class

public class TestScoresModel{

    private Student[] students;           // Array of students
    private int indexSelectedStudent;     // Position of current student
    private int studentCount;            // Current number of students

    public TestScoresModel(){

        // Initialize the data
        indexSelectedStudent = -1;
        studentCount = 0;
        students = new Student[10];
    }

    // Mutator methods for adding and replacing students

    public String add(Student s){
        if (studentCount == students.length)
            return "SORRY: student list is full";
        else{
            students[studentCount] = s;
            indexSelectedStudent = studentCount;
            studentCount++;
            return null;
        }
    }

    public String replace(Student s){
        if (indexSelectedStudent == -1)
            return "Must add a student first";
        else{
            students[indexSelectedStudent] = s;
            return null;
        }
    }

    // Navigation methods

    public Student first(){
        Student s = null;
        if (studentCount == 0)
            indexSelectedStudent = -1;
        else{
            indexSelectedStudent = 0;
            s = students[indexSelectedStudent];
        }
        return s;
    }
}

```

```

public Student previous(){
    Student s = null;
    if (studentCount == 0)
        indexSelectedStudent = -1;
    else{
        indexSelectedStudent
            = Math.max (0, indexSelectedStudent - 1);
        s = students[indexSelectedStudent];
    }
    return s;
}

public Student next(){
    Student s = null;
    if (studentCount == 0)
        indexSelectedStudent = -1;
    else{
        indexSelectedStudent
            = Math.min (studentCount - 1, indexSelectedStudent + 1);
        s = students[indexSelectedStudent];
    }
    return s;
}

public Student last(){
    Student s = null;
    if (studentCount == 0)
        indexSelectedStudent = -1;
    else{
        indexSelectedStudent = studentCount - 1;
        s = students[indexSelectedStudent];
    }
    return s;
}

// Accessors to observe data

public Student currentStudent(){
    if (indexSelectedStudent == -1)
        return null;
    else
        return students[indexSelectedStudent];
}

public int size(){
    return studentCount;
}

public int currentPosition(){
    return indexSelectedStudent;
}

```

```
public int getClassAverage(){
    if (studentCount == 0)
        return 0;
    int sum = 0;
    for (int i = 0; i < studentCount; i++)
        sum += students[i].getAverage();
    return sum / studentCount;
}

public Student getHighScore(){
    if (studentCount == 0)
        return null;
    else{
        Student s = students[0];
        for (int i = 1; i < studentCount; i++)
            if (s.getHighScore() < students[i].getHighScore())
                s = students[i];
        return s;
    }
}

public String toString(){
    String result = "";
    for (int i = 0; i < studentCount; i++)
        result = result + students[i] + "\n";
    return result;
}
}
```