```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SchoolOopData
{
    public class SchoolManagement
    {
        private List<Student> students = new List<Student>();
        private List<Teacher> teachers = new List<Teacher>();
        private List<Subject> subjects = new List<Subject>();

        public List<Student> Students
        {
            get { return students; }
            private set { students = value; }
        }

        public List<Teacher> Teachers
        {
            get { return teachers; }
            private set { teachers = value; }
        }

        public List<Subject> Subjects
        {
            get { return subjects; }
            private set { subjects = value; }
        }
        public void AddStudent(Student student)
        {
            students.Add(student);
        }
        public void AddTeacher(Teacher teacher)
        {
            teachers.Add(teacher);
        }
        public void AddSubject(Subject subject)
        {
            subjects.Add(subject);
        }
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SchoolOopData
{
    public class Teacher
    {
        public virtual int TeacherId { get; set; }
        public virtual string TeacherName { get; set; }
    }
```

```
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SchoolOopData
{
    public class Student
    {
        public virtual int StudentId { get; set; }
        public virtual string StudentName { get; set; }
        public virtual string Class { get; set; }
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SchoolOopData
{
    public class Subject
    {
        public virtual int SubjectId { get; set; }
        public virtual string SubjectName { get; set; }
    }
}
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Moq;
using NUnit.Framework;
using SchoolOopData;
using System;
using Assert = NUnit.Framework.Assert;

namespace UnitTestProject
{
    [TestFixture]
    public class SchoolUnitTest
    {
        [Test]
        public void AddStudent()
        {
            var schoolManagement = new SchoolManagement();
            var student = new Student { StudentId = 1, StudentName = "Mahesh", Class
= "Class 10" };

            // Act
            schoolManagement.AddStudent(student);

            // Assert
            Assert.Contains(student, schoolManagement.Students);
        }
        [Test]
        public void AddSubject()
        {
```

```
    var schoolManagement = new SchoolManagement();
    var subject = new Subject { SubjectId = 1, SubjectName = "Math" };

    // Act
    schoolManagement.AddSubject(subject);

    // Assert
    Assert.Contains(subject, schoolManagement.Subjects);
}
[Test]
public void AddTeacher()
{
    var schoolManagement = new SchoolManagement();
    var teacher = new Teacher { TeacherId = 1, TeacherName = "Mr.Kohli" };

    // Act
    schoolManagement.AddTeacher(teacher);

    // Assert
    Assert.Contains(teacher, schoolManagement.Teachers);
}
[Test]
public void StudentMockTest()
{
    var schoolManagement = new SchoolManagement();
    var studentMock = new Mock<Student>();
    studentMock.SetupGet(s => s.StudentId).Returns(1);
    studentMock.SetupGet(s => s.StudentName).Returns("Mahesh");
    studentMock.SetupGet(s => s.Class).Returns("Class 10");

    // Act
    schoolManagement.AddStudent(studentMock.Object);

    // Assert
    Assert.Contains(studentMock.Object, schoolManagement.Students);
}
[Test]
public void TeacherMockTest()
{
    var schoolManagement = new SchoolManagement();
    var teacherMock = new Mock<Teacher>();
    teacherMock.SetupGet(s => s.TeacherId).Returns(1);
    teacherMock.SetupGet(s => s.TeacherName).Returns("Mr.Kohli");


    // Act
    schoolManagement.AddTeacher(teacherMock.Object);

    // Assert
    Assert.Contains(teacherMock.Object, schoolManagement.Teachers);
}
[Test]
public void SubjectMockTest()
{
    var schoolManagement = new SchoolManagement();
    var subjectMock = new Mock<Subject>();
    subjectMock.SetupGet(s => s.SubjectId).Returns(1);
    subjectMock.SetupGet(s => s.SubjectName).Returns("Math");
```

```
            // Act
            schoolManagement.AddSubject(subjectMock.Object);

            // Assert
            Assert.Contains(subjectMock.Object, schoolManagement.Subjects);
        }
    }
}
```