SOURCE CODE

Create an OOP Based System for Storing School Data Using Design Patterns

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

namespace DesignProject
{

    // Singleton Pattern
    public class SchoolDataStorage
    {
        private static SchoolDataStorage instance;
        public List<Student> Students { get; set; }
        public List<Teacher> Teachers { get; set; }
        public List<Subject> Subjects { get; set; }

        private SchoolDataStorage()
        {
            Students = new List<Student>();
            Teachers = new List<Teacher>();
            Subjects = new List<Subject>();
        }

        public static SchoolDataStorage Instance
```

```
    {
        get
        {
            if (instance == null)
            {
                instance = new SchoolDataStorage();
            }
            return instance;
        }
    }
}

// Subject class
public class Subject
{
    public string Name { get; set; }
    public string SubjectCode { get; set; }
    public Teacher Teacher { get; set; }
}

// Teacher class
public class Teacher
{
    public string Name { get; set; }
    public string ClassAndSection { get; set; }
}

// Student class
public class Student
```

```
{
    public string Name { get; set; }
    public string ClassAndSection { get; set; }
}

// Repository Pattern
public class SchoolRepository
{
    private SchoolDataStorage dataStorage;

    public SchoolRepository()
    {
        dataStorage = SchoolDataStorage.Instance;
    }

    public void AddStudent(Student student)
    {
        try
        {
            dataStorage.Students.Add(student);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error adding student: {ex.Message}");
        }
    }

    public void AddTeacher(Teacher teacher)
    {
```

```
        try
        {
            dataStorage.Teachers.Add(teacher);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error adding teacher: {ex.Message}");
        }
    }

    public void AddSubject(Subject subject)
    {
        try
        {
            dataStorage.Subjects.Add(subject);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error adding subject: {ex.Message}");
        }
    }

    public List<Student> GetStudentsInClass(string classAndSection)
    {
        try
        {
            return dataStorage.Students.FindAll(student => student.ClassAndSection == classAndSection);
        }
        catch (Exception ex)
```

```
        {
            Console.WriteLine($"Error getting students in class: {ex.Message}");
            return new List<Student>();
        }
    }

    public List<Subject> GetSubjectsTaughtByTeacher(string teacherName)
    {
        try
        {
            return dataStorage.Subjects.FindAll(subject => subject.Teacher.Name == teacherName);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error getting subjects taught by teacher: {ex.Message}");
            return new List<Subject>();
        }
    }

    public void UpdateStudent(Student existingStudent, Student updatedStudent)
    {
        try
        {
            int index = dataStorage.Students.IndexOf(existingStudent);
            if (index != -1)
            {
                dataStorage.Students[index] = updatedStudent;
            }
            else
```

```
                {
                    Console.WriteLine("Student not found for updating.");
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error updating student: {ex.Message}");
            }
        }

        public void RemoveSubject(Subject subjectToRemove)
        {
            try
            {
                dataStorage.Subjects.Remove(subjectToRemove);
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error removing subject: {ex.Message}");
            }
        }
    }

    class Program
    {
        static void Main()
        {
            SchoolRepository repository = new SchoolRepository();
```

```
// Adding dummy data
repository.AddStudent(new Student { Name = "Ragendhu Ramesh", ClassAndSection = "ClassA"
});
repository.AddStudent(new Student { Name = "Gerard Joshua", ClassAndSection = "ClassA" });
repository.AddStudent(new Student { Name = "Priyanka Shivappa", ClassAndSection = "ClassA" });
repository.AddStudent(new Student { Name = "Harini Purushotham", ClassAndSection = "ClassA"
});
repository.AddStudent(new Student { Name = "Harathi V Raman", ClassAndSection = "ClassA" });
repository.AddStudent(new Student { Name = "Vyshnavi V", ClassAndSection = "ClassA" });

repository.AddStudent(new Student { Name = "Anjali P Shaji", ClassAndSection = "ClassB" });
repository.AddStudent(new Student { Name = "Neha Ravi", ClassAndSection = "ClassB" });
repository.AddStudent(new Student { Name = "Gaddam Akheel", ClassAndSection = "ClassB" });
repository.AddStudent(new Student { Name = "Rini Varghese", ClassAndSection = "ClassB" });

repository.AddTeacher(new Teacher { Name = "Teacher 1", ClassAndSection = "ClassA" });
repository.AddTeacher(new Teacher { Name = "Teacher 2", ClassAndSection = "ClassB" });

repository.AddSubject(new Subject
{
    Name = "Science",
    SubjectCode = "MATH101",
    Teacher = new Teacher { Name = "Teacher1", ClassAndSection = "ClassA" }
});

// Displaying lists
Console.WriteLine("Students in ClassA:");
foreach (var student in repository.GetStudentsInClass("ClassA"))
{
    Console.WriteLine(student.Name);
```

```
        }

        Console.WriteLine("\nSubjects taught by Teacher1:");
        foreach (var subject in repository.GetSubjectsTaughtByTeacher("Teacher1"))
        {
            Console.WriteLine(subject.Name);
        }

        repository.AddSubject(new Subject
        {
            Name = "Computer Science",
            SubjectCode = "Cs102",
            Teacher = new Teacher { Name = "Teacher2", ClassAndSection = "ClassB" }
        });

        Console.WriteLine("");
        Console.WriteLine("Students in ClassB:");
        foreach (var student in repository.GetStudentsInClass("ClassB"))
        {
            Console.WriteLine(student.Name);
        }

        Console.WriteLine("\nSubjects taught by Teacher2:");
        foreach (var subject in repository.GetSubjectsTaughtByTeacher("Teacher2"))
        {
            Console.WriteLine(subject.Name);
        }
    }
}
```

```
}
```