CTU 2023

SUBJECT NAME: Programming with C Semester 1

SUBJECT CODE: PRG521 – FA 2

Edward Nhlapo

Student Number – 20220865

20220865@ctucareer.co.za

14 September 2023:

Question 1

1.1 Write a program in LINQ and C# Sharp to find the string which starts and ends with a specific character.

Tasks to complete:

• You are to use an array that will contain 10 South African cities – You are required to use cities provided below:

Test data: Butterworth, Mthatha, Jagersfontein, Kroonstad, Boksburg, Soweto, Empangeni, Polokwane,

Secunda, Kuruman.

• Have a welcome message to your users that will help them know your application

• Display all cities available

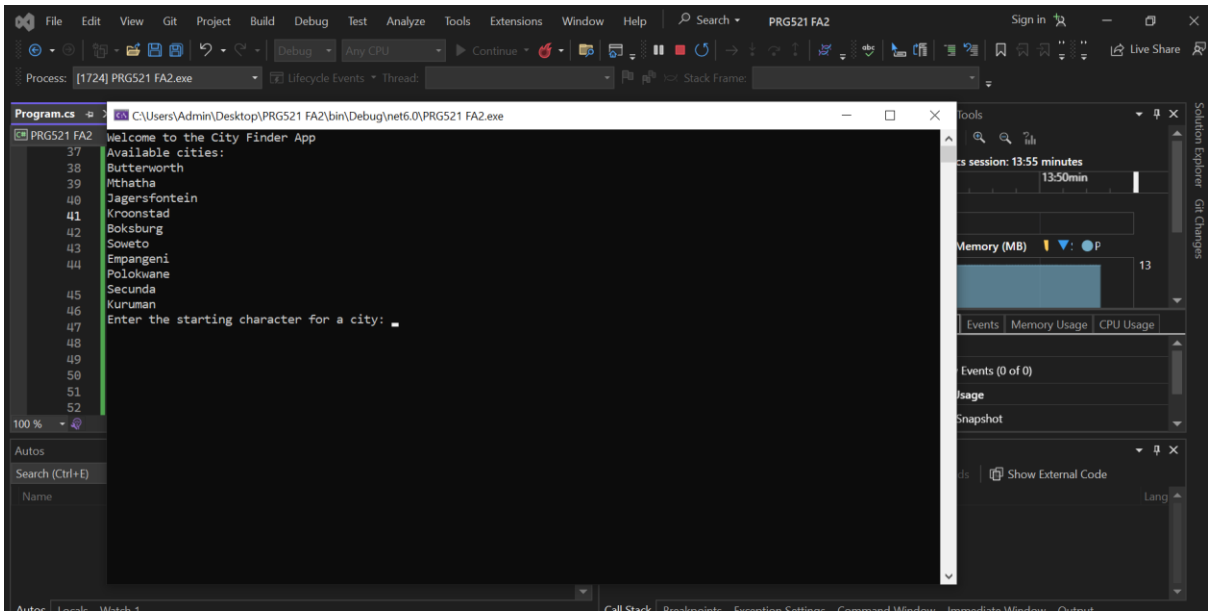
• Prompt the user to enter a starting character for a city

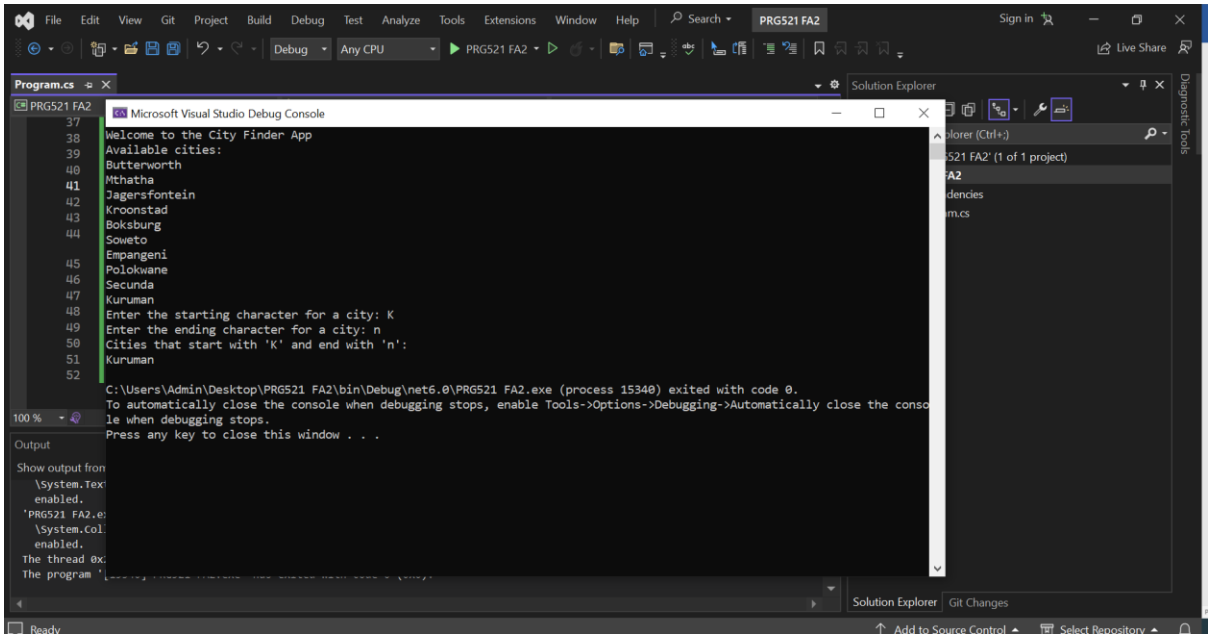
• Prompt the user to enter an ending string character for a city

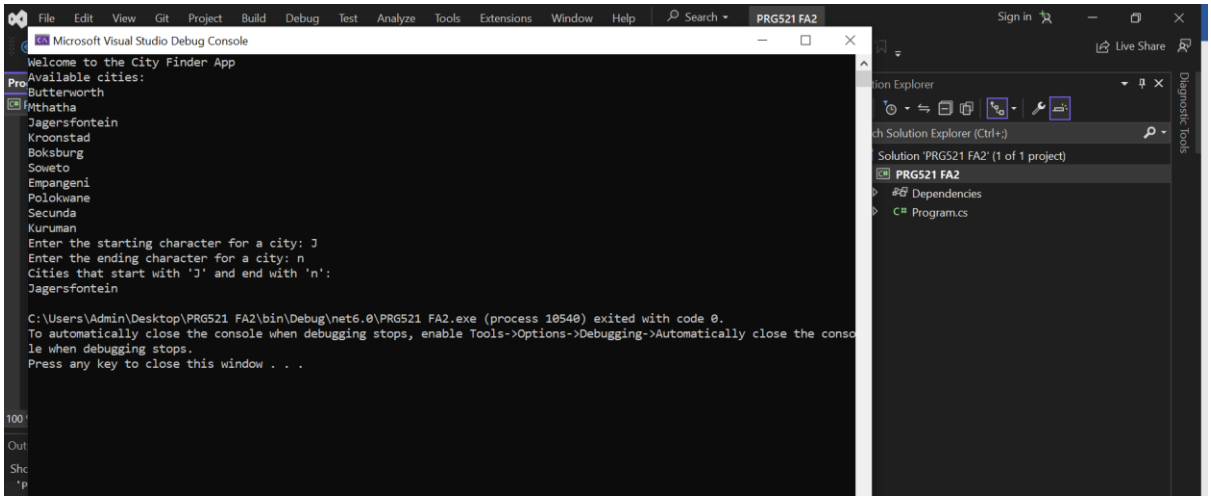
• Your output should be based on the starting and ending string character

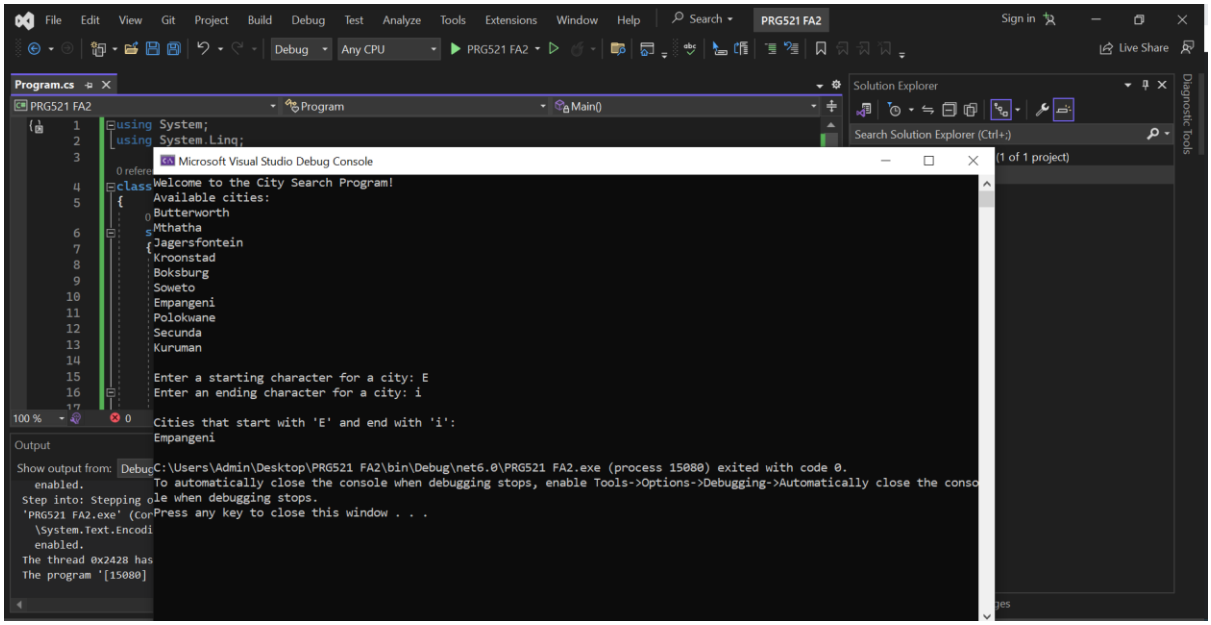
Answer:

Screen shots:




Microsoft Visual Studio Debug Console
Welcome to the City Finder App
Available cities:
Butterworth
Mthatha
Jagersfontein
Kroonstad
Boksburg
Soweto
Empangeni
Polokwane
Secunda
Kuruman
Enter the starting character for a city: K
Enter the ending character for a city: n
Cities that start with 'K' and end with 'n':
Kuruman
C:\Users\Admin\Desktop\PRG521 FA2\bin\Debug\net6.0\PRG521 FA2.exe (process 15340) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .


Microsoft Visual Studio Debug Console
Welcome to the City Finder App
Available cities:
Butterworth
Mthatha
Jagersfontein
Kroonstad
Boksburg
Soweto
Empangeni
Polokwane
Secunda
Kuruman
Enter the starting character for a city: J
Enter the ending character for a city: n
Cities that start with 'J' and end with 'n':
Jagersfontein
C:\Users\Admin\Desktop\PRG521 FA2\bin\Debug\net6.0\PRG521 FA2.exe (process 10540) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .



Code:

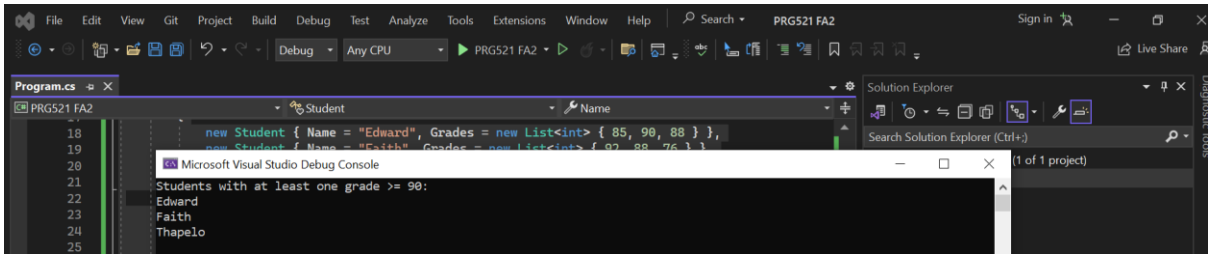
```
using System;
using System.Linq;

class Program
{
    static void Main()
    {
        // Array of South African cities
        string[] cities = {
            "Butterworth", "Mthatha", "Jagersfontein", "Kroonstad", "Boksburg",
            "Soweto", "Empangeni", "Polokwane", "Secunda", "Kuruman"
        };

        Console.WriteLine("Welcome to the City Finder App");
        Console.WriteLine("Available cities:");
        DisplayCities(cities);

        Console.Write("Enter the starting character for a city: ");
        char startChar = Console.ReadLine().FirstOrDefault();

        Console.Write("Enter the ending character for a city: ");
        char endChar = Console.ReadLine().FirstOrDefault();

        var matchingCities = FindCities(cities, startChar, endChar);

        if (matchingCities.Any())
        {
            Console.WriteLine($"Cities that start with '{startChar}' and end with '{endChar}':");
            DisplayCities(matchingCities);
        }
```

```
        else
        {
            Console.WriteLine("No matching cities found.");
        }
    }

    static void DisplayCities(string[] cities)
    {
        foreach (var city in cities)
        {
            Console.WriteLine(city);
        }
    }

    static string[] FindCities(string[] cities, char startChar, char endChar)
    {
        return cities
            .Where(city => city.Length > 0 && city[0] == startChar && city[city.Length - 1] == endChar)
            .ToArray();
    }
}
```

<u>Question 2</u>.

Write a LINQ query that retrieves the names of all the students who have at least one grade greater than or equal

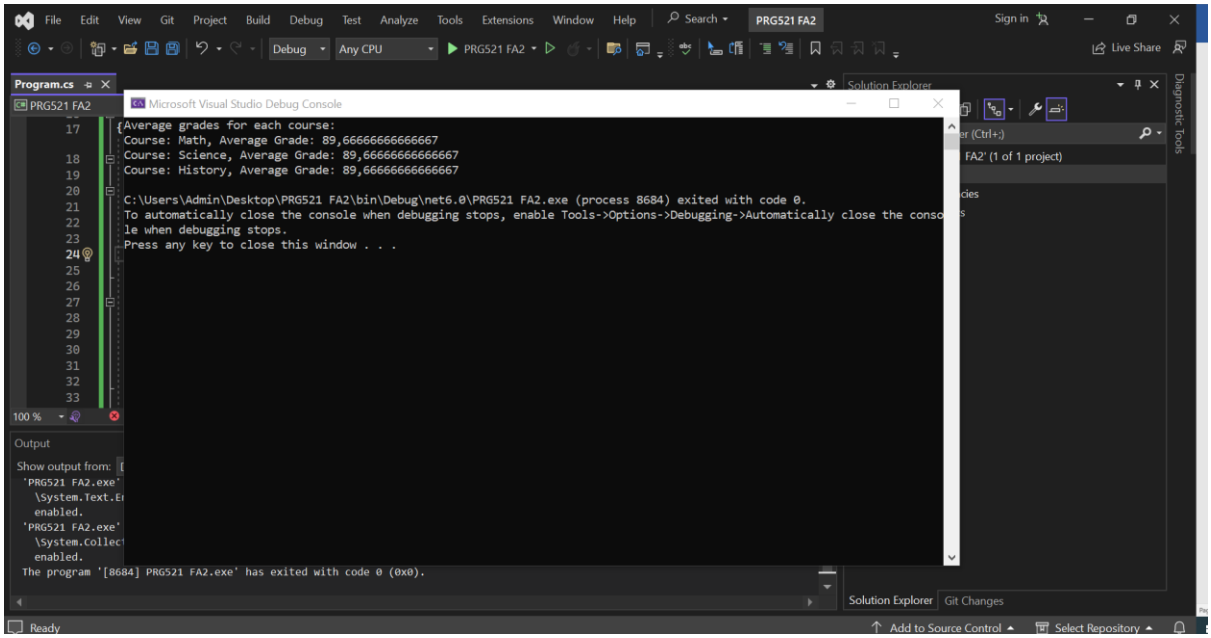
to 90.




```
using System;

using System.Collections.Generic;

using System.Linq;


class Student
```

```
{
    public string Name { get; set; }
    public List<int> Grades { get; set; }
}

class Program
{
    static void Main()
    {
        // Sample data
        List<Student> students = new List<Student>
        {
            new Student { Name = "Edward", Grades = new List<int> { 85, 90, 88 } },
            new Student { Name = "Faith", Grades = new List<int> { 92, 88, 76 } },
            new Student { Name = "Thapelo", Grades = new List<int> { 95, 89, 92 } }
        };

        // LINQ query to retrieve names of students with at least one grade >= 90
        var highScoringStudents = students
            .Where(student => student.Grades.Any(grade => grade >= 90))
            .Select(student => student.Name);

        Console.WriteLine("Students with at least one grade >= 90:");
        foreach (var studentName in highScoringStudents)
        {
            Console.WriteLine(studentName);
        }
    }
}
```

Write a LINQ query that calculates the average grade of all the students in each course, and returns a list of anonymous objects with the course name and the average grade

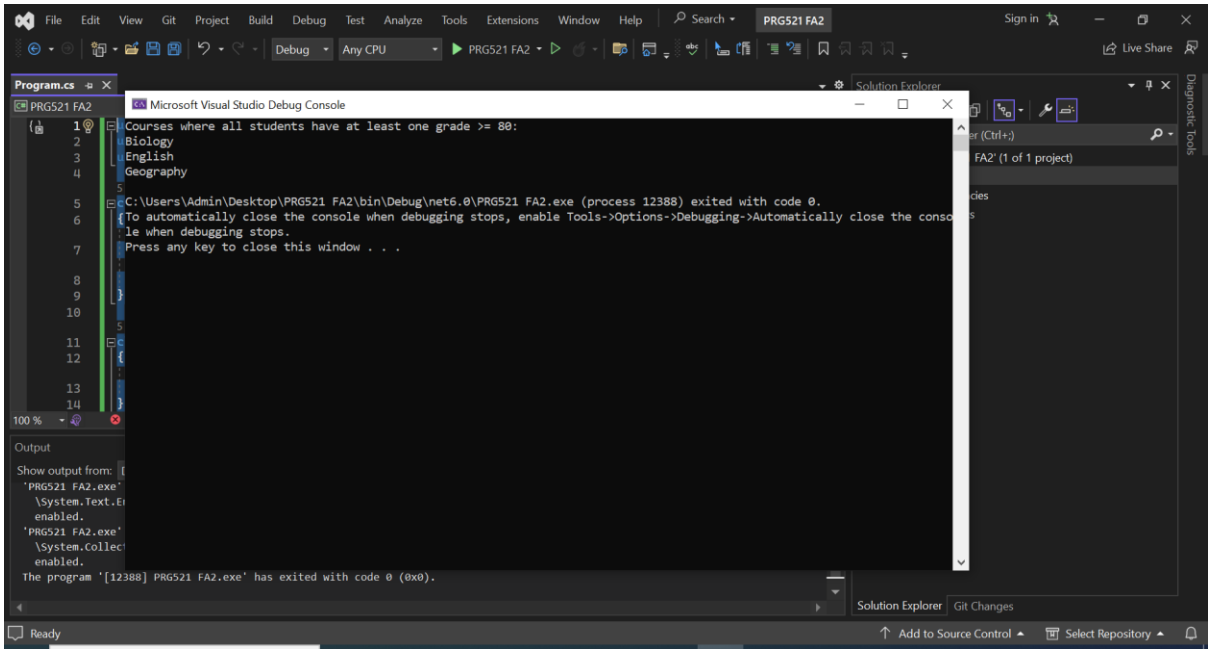



```
using System;
using System.Collections.Generic;
using System.Linq;

class Student
{
    public string Name { get; set; }
    public List<int> Grades { get; set; }
}

class Course
{
    public string Name { get; set; }
}

class Program
{
    static void Main()
    {
        List<Student> students = new List<Student>
        {
            new Student { Name = "Edward", Grades = new List<int> { 87, 96, 84 } },
            new Student { Name = "Faith", Grades = new List<int> { 94, 89, 77 } },
            new Student { Name = "Sikhumbuzo", Grades = new List<int> { 97, 90, 93 } }
        };
```

```
        List<Course> courses = new List<Course>
        {
            new Course { Name = "Math" },
            new Course { Name = "Science" },
            new Course { Name = "History" }
        };

        // LINQ query to calculate average grade for each course
        var courseAverages = courses
            .Select(course => new
            {
                CourseName = course.Name,
                AverageGrade = students
                    .SelectMany(student => student.Grades) // Flatten grades across all students
                    .Average() // Calculate the average grade
            });

        Console.WriteLine("Average grades for each course:");
        foreach (var courseAverage in courseAverages)
        {
            Console.WriteLine($"Course: {courseAverage.CourseName}, Average Grade:
{courseAverage.AverageGrade}");
        }
    }
}
```

3- Write a LINQ query that retrieves the names of all the courses where all the students have at least one grade

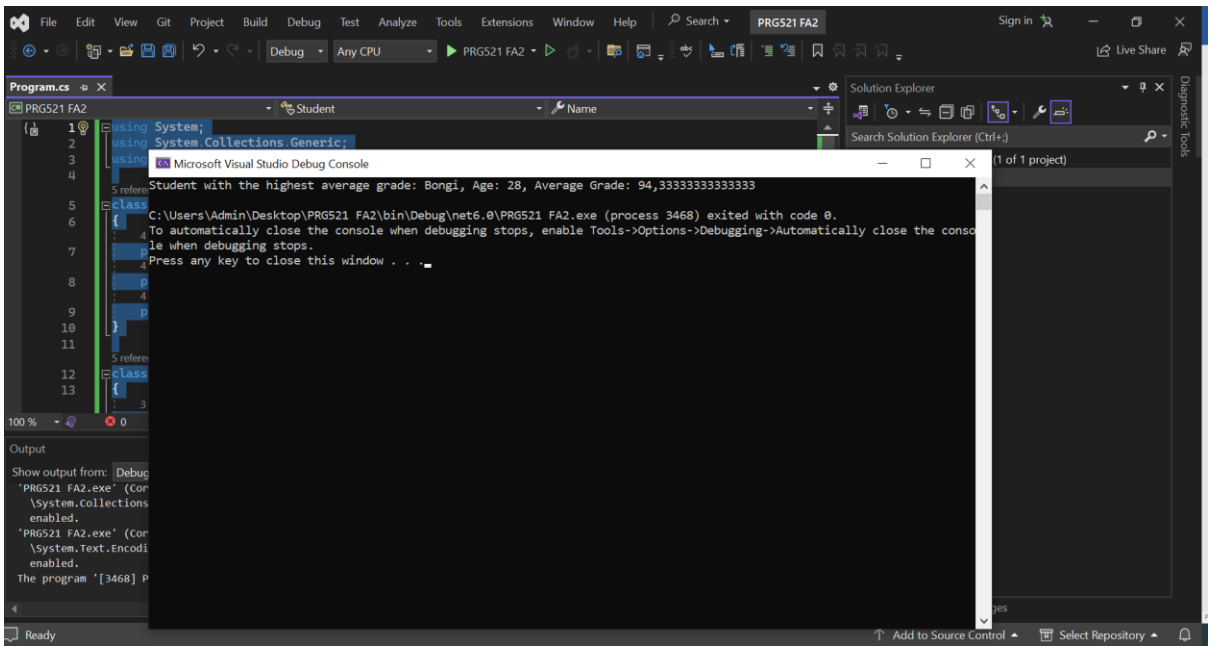
greater than or equal to 80.




```
using System;
using System.Collections.Generic;
using System.Linq;

class Student
{
    public string Name { get; set; }
    public List<int> Grades { get; set; }
}

class Course
{
    public string Name { get; set; }
}

class Program
{
    static void Main()
    {
        List<Student> students = new List<Student>
        {
            new Student { Name = "Edward", Grades = new List<int> { 85, 90, 88 } },
            new Student { Name = "Duduzile", Grades = new List<int> { 92, 88, 76 } },
            new Student { Name = "Carol", Grades = new List<int> { 95, 89, 92 } }
        };

        List<Course> courses = new List<Course>
        {
            new Course { Name = "Biology" },
            new Course { Name = "English" },
```

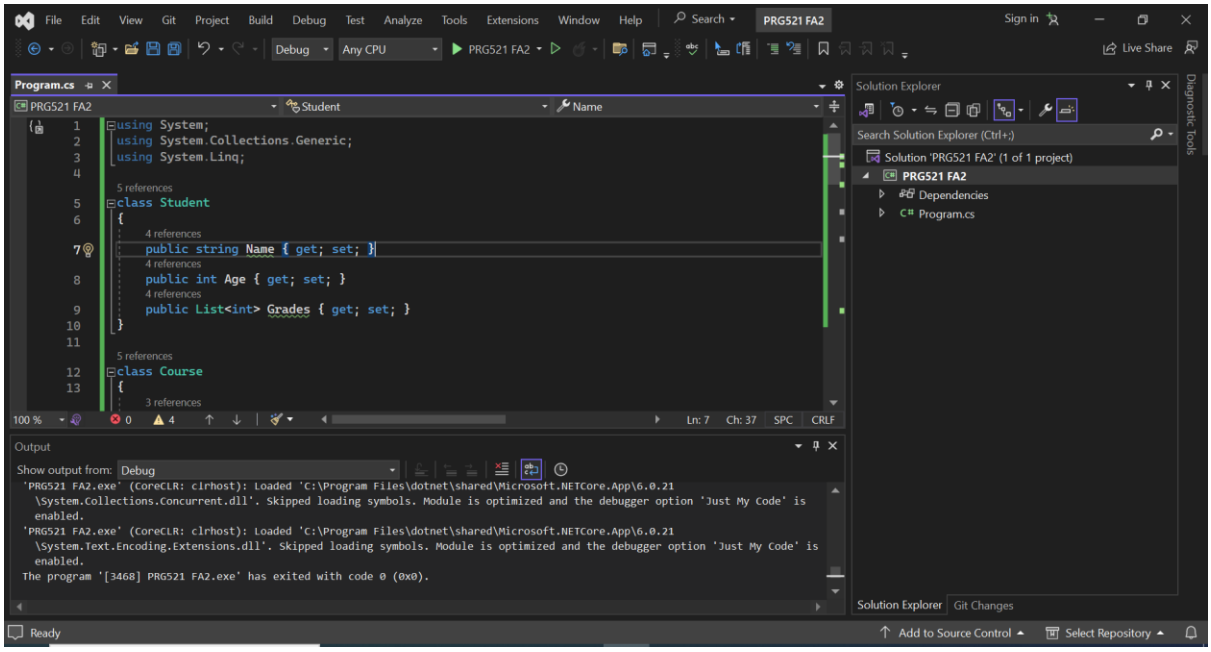
```
        new Course { Name = "Geography" }
    };

    // LINQ query to retrieve courses where all students have at least one grade >= 80
    var coursesWithHighGrades = courses
        .Where(course => students.All(student => student.Grades.Any(grade => grade >= 80)))
        .Select(course => course.Name);

    Console.WriteLine("Courses where all students have at least one grade >= 80:");
    foreach (var courseName in coursesWithHighGrades)
    {
        Console.WriteLine(courseName);
    }
  }
}
```

- 4 Write a LINQ query that retrieves the name and age of the student with the highest average grade across all the courses.




```
using System;
using System.Collections.Generic;
using System.Linq;

class Student
{
    public string Name { get; set; }
    public int Age { get; set; }
```

```
    public List<int> Grades { get; set; }
}

class Course
{
    public string Name { get; set; }
}

class Program
{
    static void Main()
    {
        List<Student> students = new List<Student>
        {
            new Student { Name = "Edward", Age = 26, Grades = new List<int> { 86, 94, 89 } },
            new Student { Name = "Mangi", Age = 25, Grades = new List<int> { 91, 84, 77 } },
            new Student { Name = "Bongi", Age = 28, Grades = new List<int> { 97, 90, 96 } }
        };

        List<Course> courses = new List<Course>
        {
            new Course { Name = "Math" },
            new Course { Name = "Science" },
            new Course { Name = "History" }
        };

        // LINQ query to find the student with the highest average grade across all courses
        var topStudent = students
            .Select(student => new
            {
                student.Name,
                student.Age,
                AverageGrade = student.Grades.Average()
            })
            .OrderByDescending(student => student.AverageGrade)
            .FirstOrDefault();

        Console.WriteLine($"Student with the highest average grade: {topStudent.Name}, Age:
{topStudent.Age}, Average Grade: {topStudent.AverageGrade}");
    }
}
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
PRG521 FA2
Sign in
Live Share
Debug
Any CPU
PRG521 FA2
Program.cs
PRG521 FA2
Student
Name
using System;
using System.Collections.Generic;
using System.Linq;
5 references
class Student
{
4 references
public string Name { get; set; }
4 references
public int Age { get; set; }
4 references
public List<int> Grades { get; set; }
}
5 references
class Course
{
3 references
100 %
0
4
Ln: 7
Ch: 37
SPC
CRLF
Output
Show output from: Debug
'PRG521 FA2.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.21
\System.Collections.Concurrent.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My Code' is
enabled.
'PRG521 FA2.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.21
\System.Text.Encoding.Extensions.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My Code' is
enabled.
The program '[3468] PRG521 FA2.exe' has exited with code 0 (0x0).
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'PRG521 FA2' (1 of 1 project)
PRG521 FA2
Dependencies
Program.cs
Diagnostic Tools
Solution Explorer
Git Changes
Ready
Add to Source Control
Select Repository