Faculty of Information Technology

SUBJECT NAME: Programming with C#
SUBJECT CODE:PRG521

I declare that I am familiar with, and will abide to the Examination rules of CTU.

Murray

Signature

Formative Assessment 2

Duration:12-26 April

Date:2023/04/21

Total Marks: 60

Total pages:24

Examiner: Mr. Junior Manganyi

Moderator:

Student number 20231805

Surname: Murray

Initials:MAK

/ %

Contents

Question 1 3

The code for Question 1 4

The screenshots of the following code in an IDE(Visual Studio) 6

The Output of the Code 8

Question 2 10

Code without the LINQ query 10

Question 2.1 12

The code with the LINQ Query 12

The output with the LINQ query (Application used: Visual Studio 14

Question 2.2 15

The code for Q 2.2 15

Code output for Q2.2 17

Question 2.3 18

The code with Q 2.3 18

The output for Q 2.3 20

Question 2 .4 21

The code for Q 2.4 21

The code output for Q 2.4 23

Declaration 24

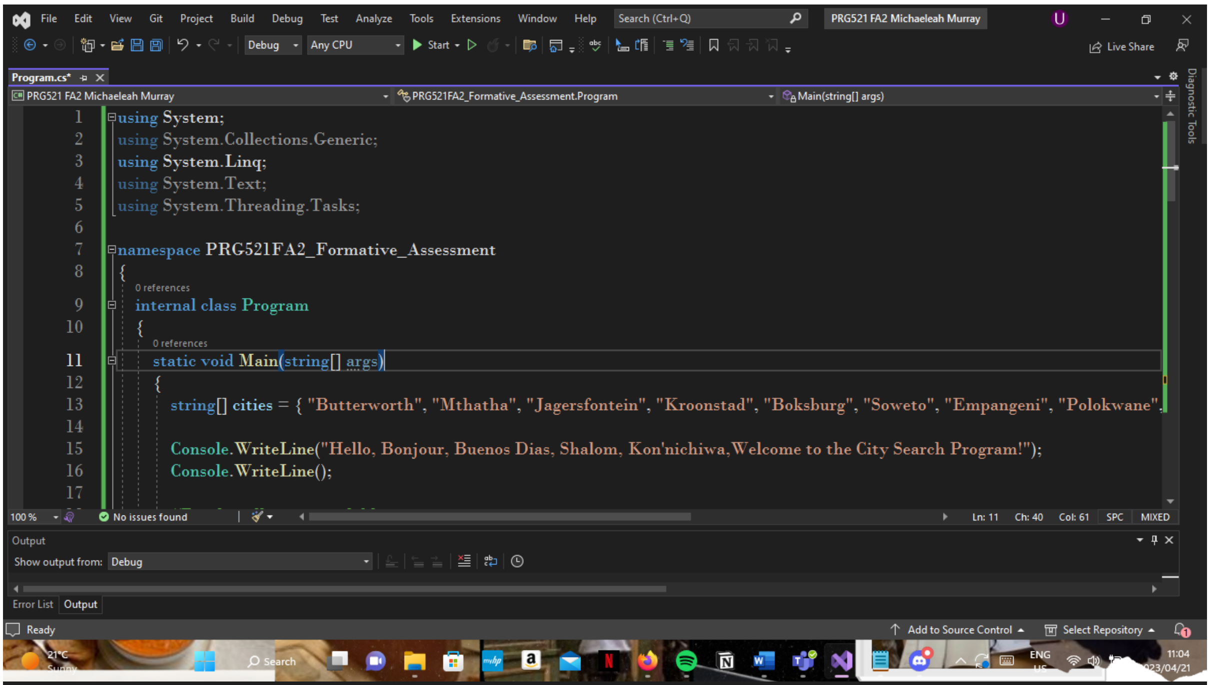
Question 1

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

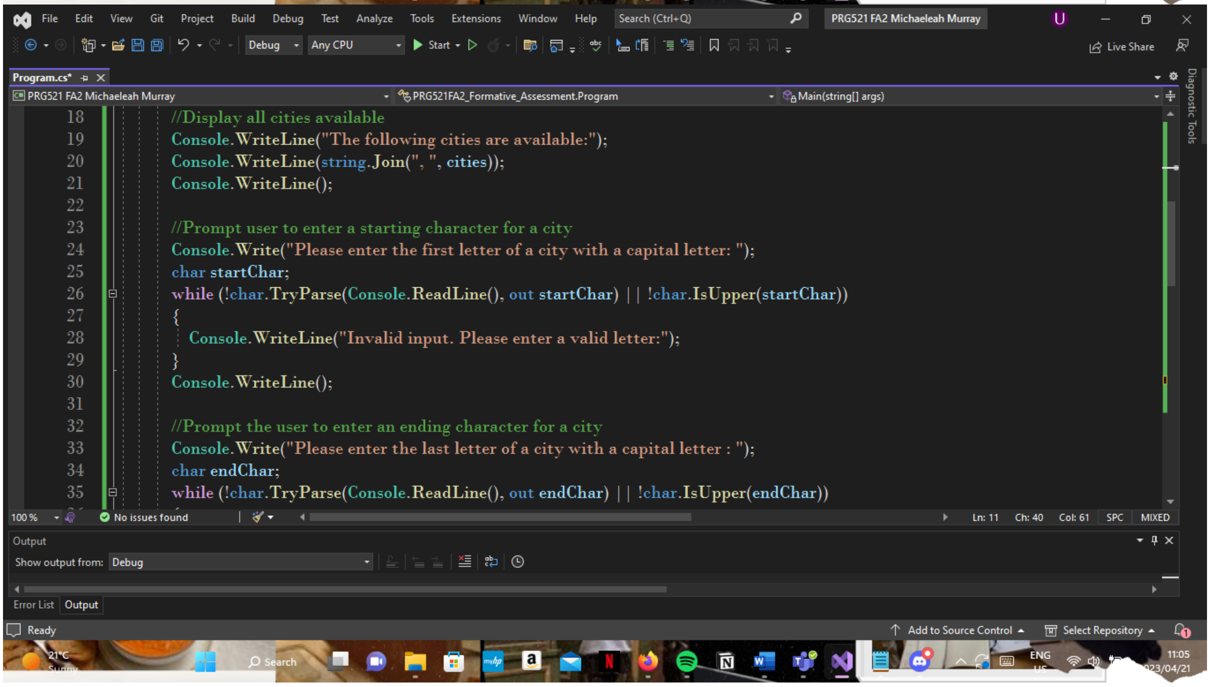
The code for Question 1

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

namespace PRG521FA2_Formative_Assessment
{
    internal class Program
    {
        static void Main(string[] args)
        {
            string[] cities = { "Butterworth", "Mthatha", "Jagersfontein",
"Kroonstad", "Boksburg", "Soweto", "Empangeni", "Polokwane", "Secunda",
"Kuruman" };

            Console.WriteLine("Hello, Bonjour, Buenos Dias, Shalom,
Kon'nichiwa,Welcome to the City Search Program!");
            Console.WriteLine();

            //Display all cities available
            Console.WriteLine("The following cities are available:");
            Console.WriteLine(string.Join(", ", cities));
            Console.WriteLine();

            //Prompt user to enter a starting character for a city
            Console.Write("Please enter the first letter of a city with a capital letter:
");
            char startChar;
            while (!char.TryParse(Console.ReadLine(), out startChar) ||
!char.IsUpper(startChar))
            {
                Console.WriteLine("Invalid input. Please enter a valid letter:");
            }
            Console.WriteLine();

            //Prompt the user to enter an ending character for a city
            Console.Write("Please enter the last letter of a city with a capital letter :
");
            char endChar;
            while (!char.TryParse(Console.ReadLine(), out endChar) ||
!char.IsUpper(endChar))
            {
                Console.WriteLine("Invalid input. Please enter a valid letter:");
            }
            Console.WriteLine();

            //Use LINQ to select cities that start with the entered starting character
and end with the entered ending character
            var result = cities.Where(city => city.StartsWith(startChar.ToString(),
StringComparison.OrdinalIgnoreCase) && city.EndsWith(endChar.ToString(),
StringComparison.OrdinalIgnoreCase));

            //Display the resulting cities to the user
            Console.WriteLine($"The following cities start with '{startChar}' and
end with '{endChar}':");
            if (result.Any())
            {
                Console.WriteLine(string.Join(", ", result));
            }
            else
            {
                Console.WriteLine("No cities were found matching the specified
criteria.");
            }

            Console.WriteLine();
            Console.WriteLine("Thank you for using the City Search Program,good
bye,au revoir,adios,totsiens,alwadae!");
            Console.ReadKey();
        }
    }
}
```

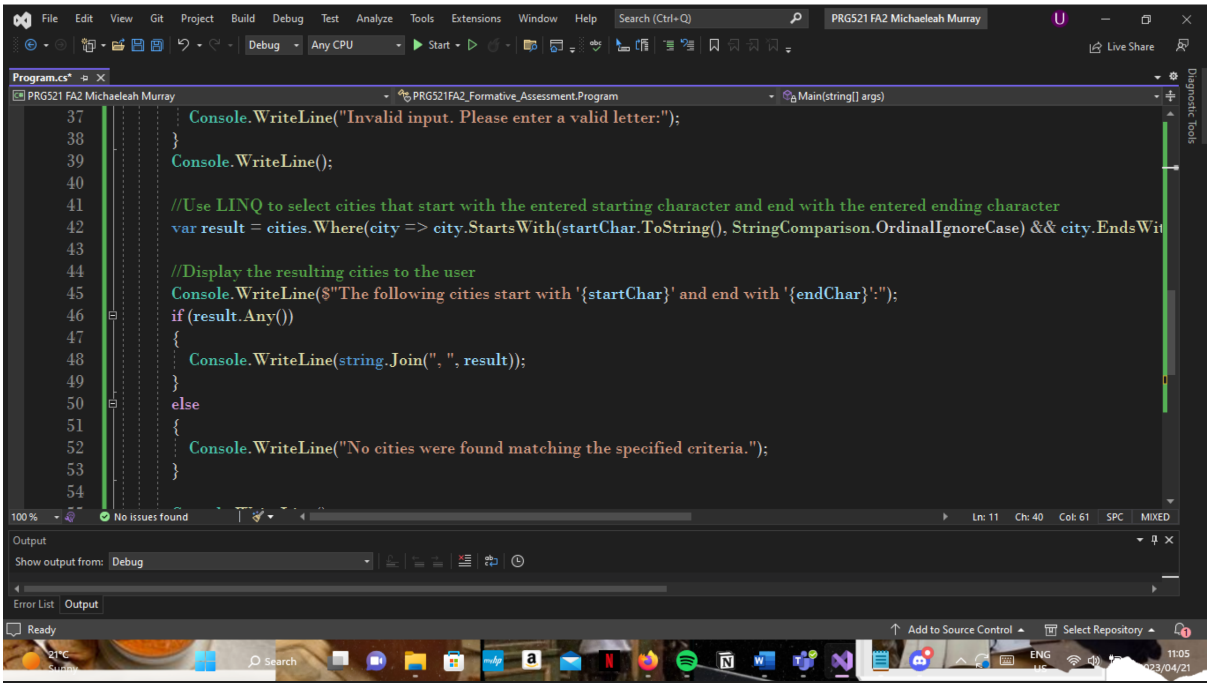
<u>The screenshots of the following code in an IDE(Visual Studio)</u>


PRG521 FA2 Michaeleah Murray
Program.cs*
PRG521FA2_Formative_Assessment.Program
Main(string[] args)
No issues found
Ln: 11 Ch: 40 Col: 61 SPC MIXED
Output
Show output from: Debug
Error List Output
Ready

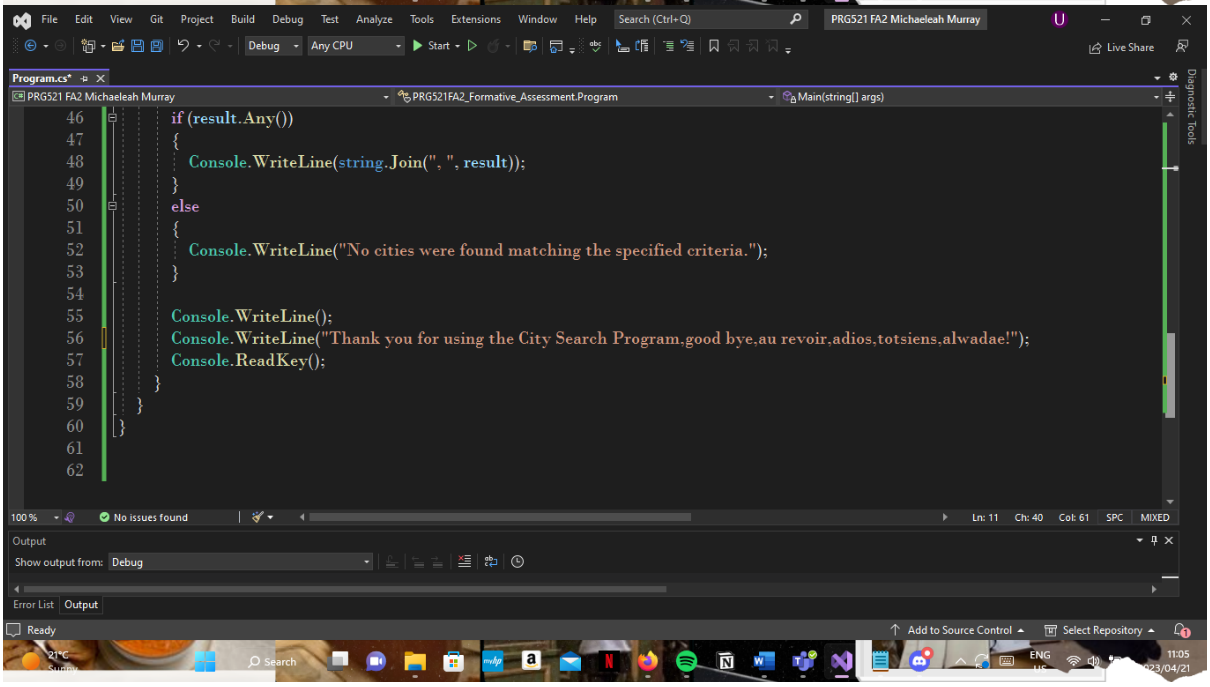
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace PRG521FA2_Formative_Assessment
{
    internal class Program
    {
        static void Main(string[] args)
        {
            string[] cities = { "Butterworth", "Mthatha", "Jagersfontein", "Kroonstad", "Boksburg", "Soweto", "Empangeni", "Polokwane",

            Console.WriteLine("Hello, Bonjour, Buenos Dias, Shalom, Kon'nichiwa,Welcome to the City Search Program!");
            Console.WriteLine();

```


PRG521 FA2 Michaeleah Murray
Program.cs*
PRG521FA2_Formative_Assessment.Program
Main(string[] args)
No issues found
Ln: 11 Ch: 40 Col: 61 SPC MIXED
Output
Show output from: Debug
Error List Output
Ready

```
            //Display all cities available
            Console.WriteLine("The following cities are available:");
            Console.WriteLine(string.Join(", ", cities));
            Console.WriteLine();

            //Prompt user to enter a starting character for a city
            Console.Write("Please enter the first letter of a city with a capital letter: ");
            char startChar;
            while (!char.TryParse(Console.ReadLine(), out startChar) || !char.IsUpper(startChar))
            {
                Console.WriteLine("Invalid input. Please enter a valid letter:");
            }
            Console.WriteLine();

            //Prompt the user to enter an ending character for a city
            Console.Write("Please enter the last letter of a city with a capital letter : ");
            char endChar;
            while (!char.TryParse(Console.ReadLine(), out endChar) || !char.IsUpper(endChar))
```

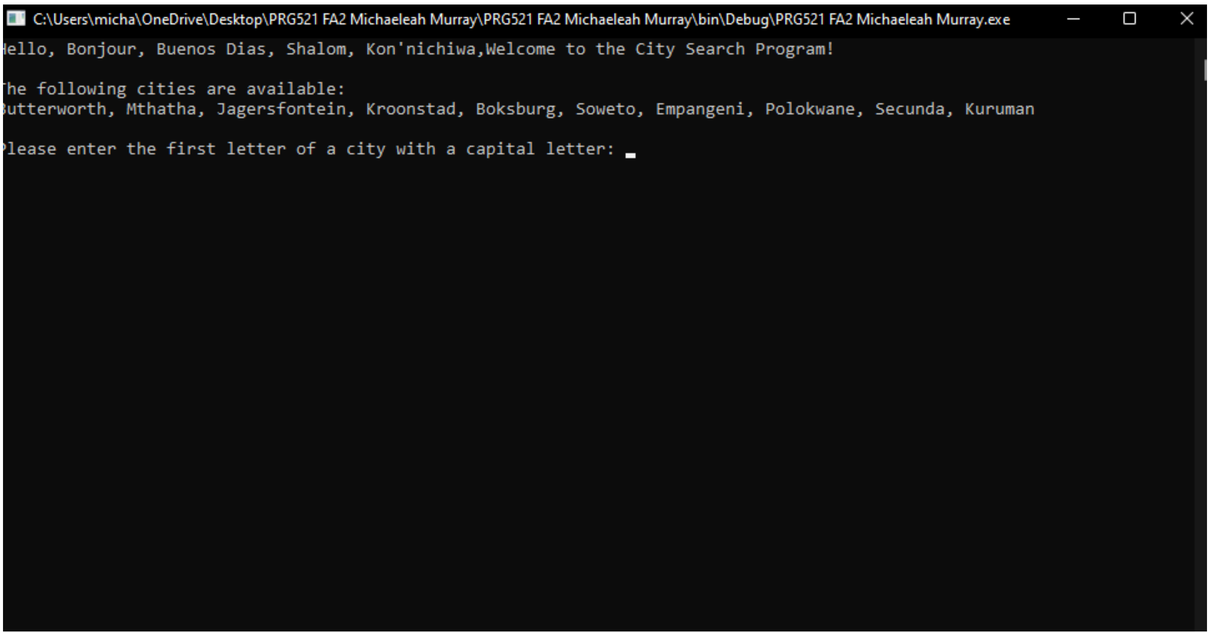


```
                Console.WriteLine("Invalid input. Please enter a valid letter:");
            }
            Console.WriteLine();

            //Use LINQ to select cities that start with the entered starting character and end with the entered ending character
            var result = cities.Where(city => city.StartsWith(startChar.ToString(), StringComparison.OrdinalIgnoreCase) && city.EndsWi

            //Display the resulting cities to the user
            Console.WriteLine($"The following cities start with '{startChar}' and end with '{endChar}':");
            if (result.Any())
            {
                Console.WriteLine(string.Join(", ", result));
            }
            else
            {
                Console.WriteLine("No cities were found matching the specified criteria.");
            }

```



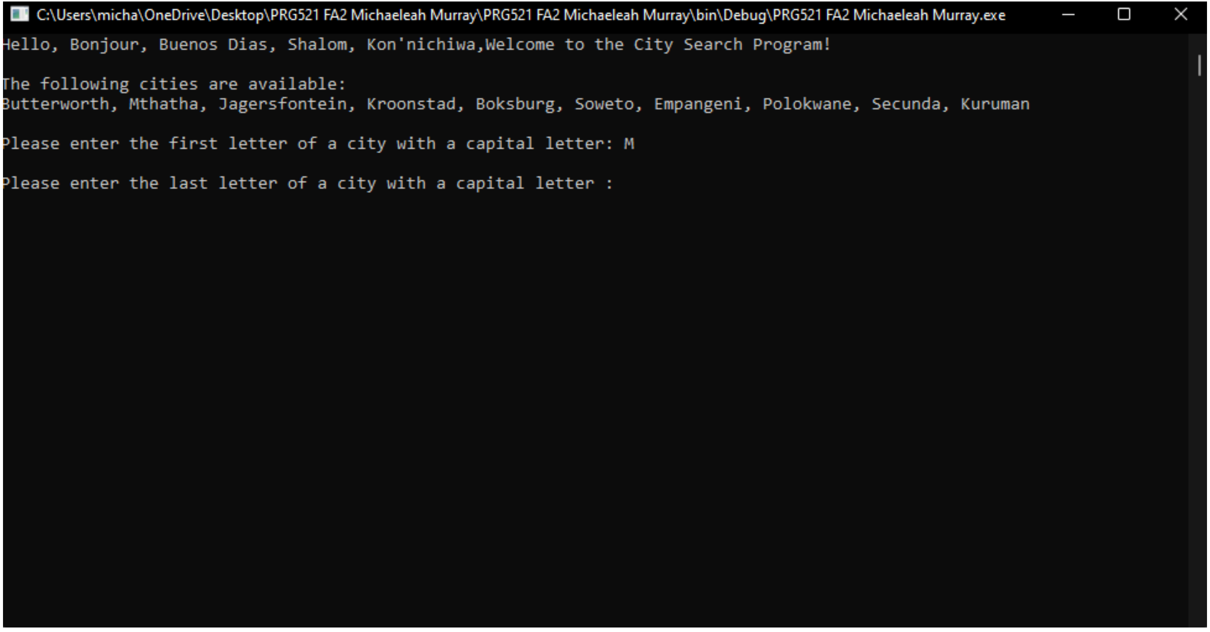
```
            if (result.Any())
            {
                Console.WriteLine(string.Join(", ", result));
            }
            else
            {
                Console.WriteLine("No cities were found matching the specified criteria.");
            }

            Console.WriteLine();
            Console.WriteLine("Thank you for using the City Search Program,good bye,au revoir,adios,totsiens,alwadae!");
            Console.ReadKey();
        }
    }
}
```

The Output of the Code



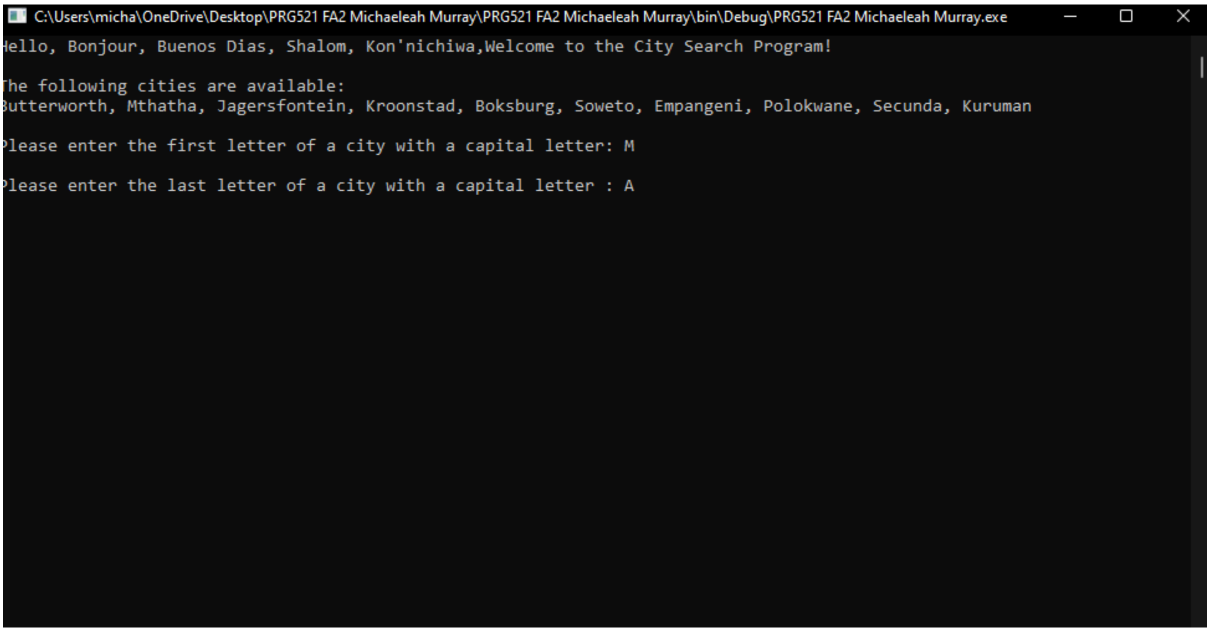
C:\Users\micha\OneDrive\Desktop\PRG521 FA2 Michaeleah Murray\PRG521 FA2 Michaeleah Murray\bin\Debug\PRG521 FA2 Michaeleah Murray.exe

```
Hello, Bonjour, Buenos Dias, Shalom, Kon'nichiwa,Welcome to the City Search Program!

The following cities are available:
Butterworth, Mthatha, Jagersfontein, Kroonstad, Boksburg, Soweto, Empangeni, Polokwane, Secunda, Kuruman

Please enter the first letter of a city with a capital letter: _
```

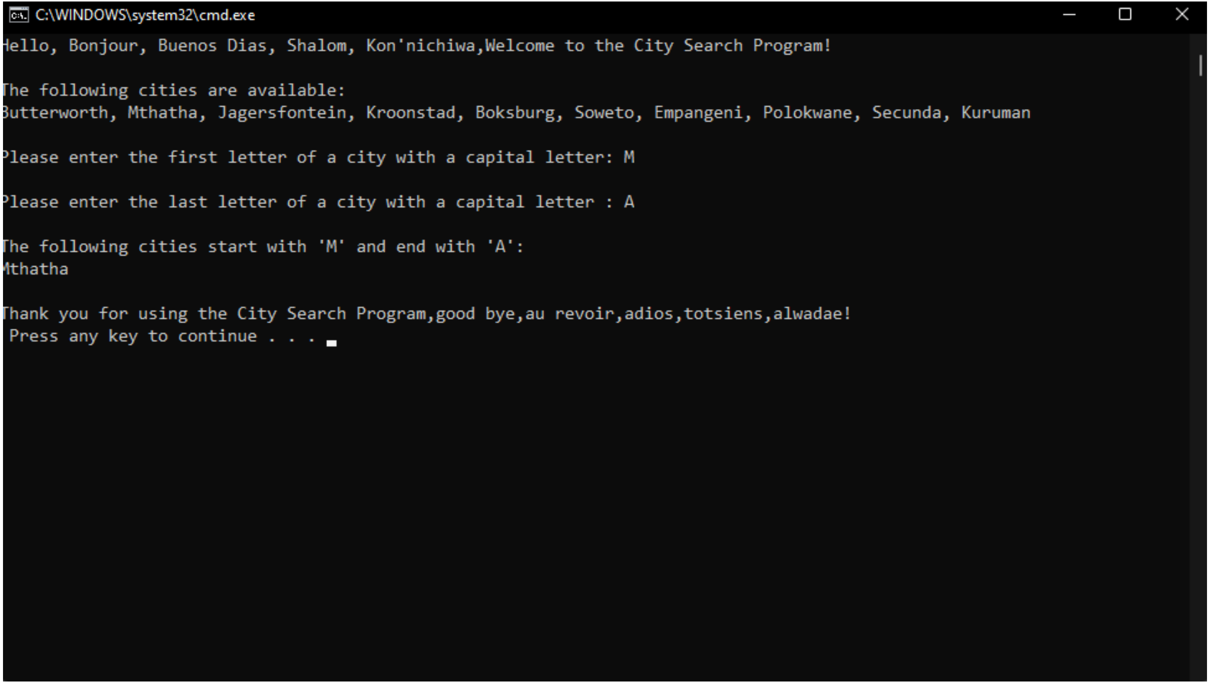


C:\Users\micha\OneDrive\Desktop\PRG521 FA2 Michaeleah Murray\PRG521 FA2 Michaeleah Murray\bin\Debug\PRG521 FA2 Michaeleah Murray.exe

```
Hello, Bonjour, Buenos Dias, Shalom, Kon'nichiwa,Welcome to the City Search Program!

The following cities are available:
Butterworth, Mthatha, Jagersfontein, Kroonstad, Boksburg, Soweto, Empangeni, Polokwane, Secunda, Kuruman

Please enter the first letter of a city with a capital letter: M

Please enter the last letter of a city with a capital letter :
```


C:\Users\micha\OneDrive\Desktop\PRG521 FA2 Michaeleah Murray\PRG521 FA2 Michaeleah Murray\bin\Debug\PRG521 FA2 Michaeleah Murray.exe

```
Hello, Bonjour, Buenos Dias, Shalom, Kon'nichiwa,Welcome to the City Search Program!

The following cities are available:
Butterworth, Mthatha, Jagersfontein, Kroonstad, Boksburg, Soweto, Empangeni, Polokwane, Secunda, Kuruman

Please enter the first letter of a city with a capital letter: M

Please enter the last letter of a city with a capital letter : A
```



```
Hello, Bonjour, Buenos Dias, Shalom, Kon'nichiwa,Welcome to the City Search Program!

The following cities are available:
Butterworth, Mthatha, Jagersfontein, Kroonstad, Boksburg, Soweto, Empangeni, Polokwane, Secunda, Kuruman

Please enter the first letter of a city with a capital letter: M

Please enter the last letter of a city with a capital letter : A

The following cities start with 'M' and end with 'A':
Mthatha

Thank you for using the City Search Program,good bye,au revoir,adios,totsiens,alwadae!
 Press any key to continue . . .
```

Question 2

Below you have been provided with lines of codes. Answer the questions based on the line of codes. You are required to recreate it and add your line of code based on the questions below and submit the complete working code together with the screenshots in PDF documents for all your outputs.

Code without the LINQ query

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

namespace student
{
    public class Student
    {
        public string Name { get; set; }
        public int Age { get; set; }
        public int[] Grades { get; set; }
    }

    public class Course
    {
        public string Name { get; set; }
        public List<Student> Students { get; set; }
    }

    class Program
    {
        static void Main(string[] args)
        {
            List<Course> courses = new List<Course>
            {
                new Course
                {
                    Name = "Math",
                    Students = new List<Student>
                    {
                        new Student { Name = "Alice", Age = 22, Grades = new int[] {
80, 85, 90 } },
                        new Student { Name = "Bob", Age = 21, Grades = new int[] {
75, 80, 85 } },
                        new Student { Name = "Charlie", Age = 23, Grades = new int[] {
90, 95, 100 } }
                    }
                },
                new Course
                {
                    Name = "English",
                    Students = new List<Student>
                    {
                        new Student { Name = "David", Age = 20, Grades = new int[] {
85, 90, 95 } },
                        new Student { Name = "Emma", Age = 22, Grades = new int[] {
95, 100, 100 } },
                        new Student { Name = "Frank", Age = 21, Grades = new int[] {
80, 85, 90 } }
                    }
                }
            };
```

Question 2.1

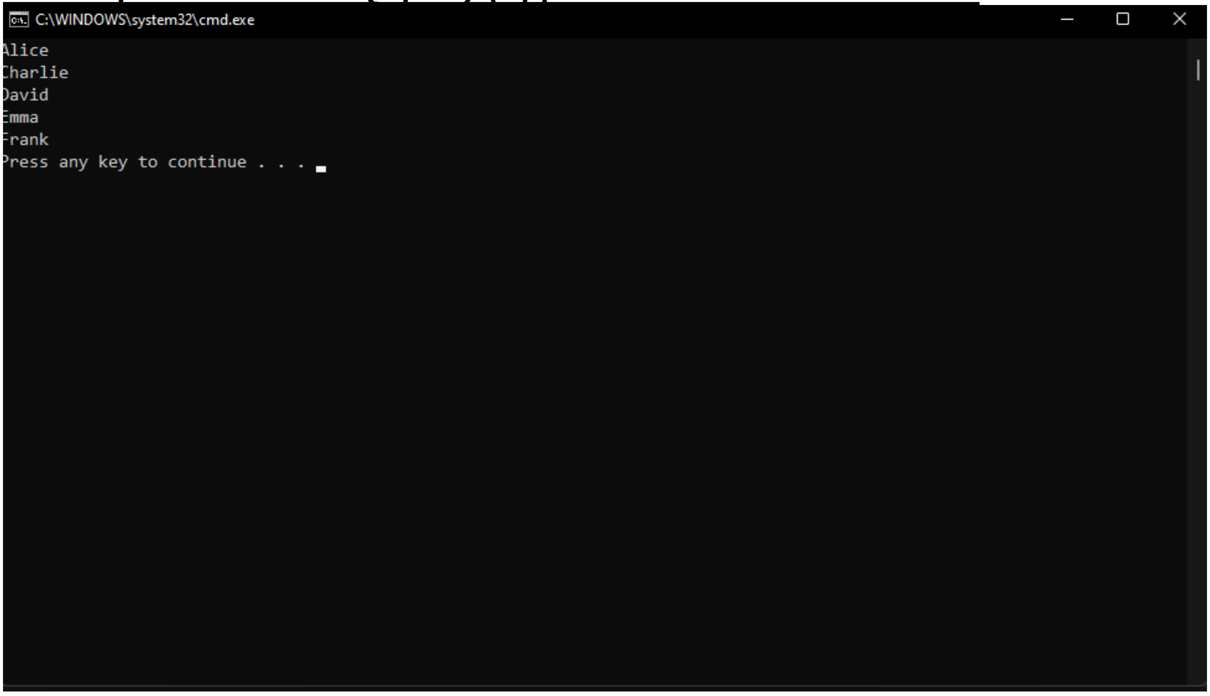
Write a LINQ query that retrieves the names of all the students who have at least one grade greater than or equal to 90.

The code with the LINQ Query

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

namespace student
{
    public class Student
    {
        public string Name { get; set; }
        public int Age { get; set; }
        public int[] Grades { get; set; }
    }

    public class Course
    {
        public string Name { get; set; }
        public List<Student> Students { get; set; }
    }

    class Program
    {
        static void Main(string[] args)
        {
            List<Course> courses = new List<Course>
            {
                new Course
                {
                    Name = "Math",
                    Students = new List<Student>
                    {
                        new Student { Name = "Alice", Age = 22, Grades = new int[] { 80, 85, 90 } },
                        new Student { Name = "Bob", Age = 21, Grades = new int[] { 75, 80, 85 } },
                        new Student { Name = "Charlie", Age = 23, Grades = new int[] {
```

```
90, 95, 100 } }
                    }
                },
                new Course
                {
                    Name = "English",
                    Students = new List<Student>
                    {
                        new Student { Name = "David", Age = 20, Grades = new int[] {
85, 90, 95 } },
                        new Student { Name = "Emma", Age = 22, Grades = new int[] {
95, 100, 100 } },
                        new Student { Name = "Frank", Age = 21, Grades = new int[] {
80, 85, 90 } }
                    }
                }
            };

            var query = from course in courses
                        from student in course.Students
                        where student.Grades.Any(g => g >= 90)
                        select student.Name;

            foreach (var name in query)
            {
                Console.WriteLine(name);
            }
        }
    }
}
```

The output with the LINQ query (Application used: Visual Studio



```
Alice
Charlie
David
Emma
Frank
Press any key to continue . . .
```

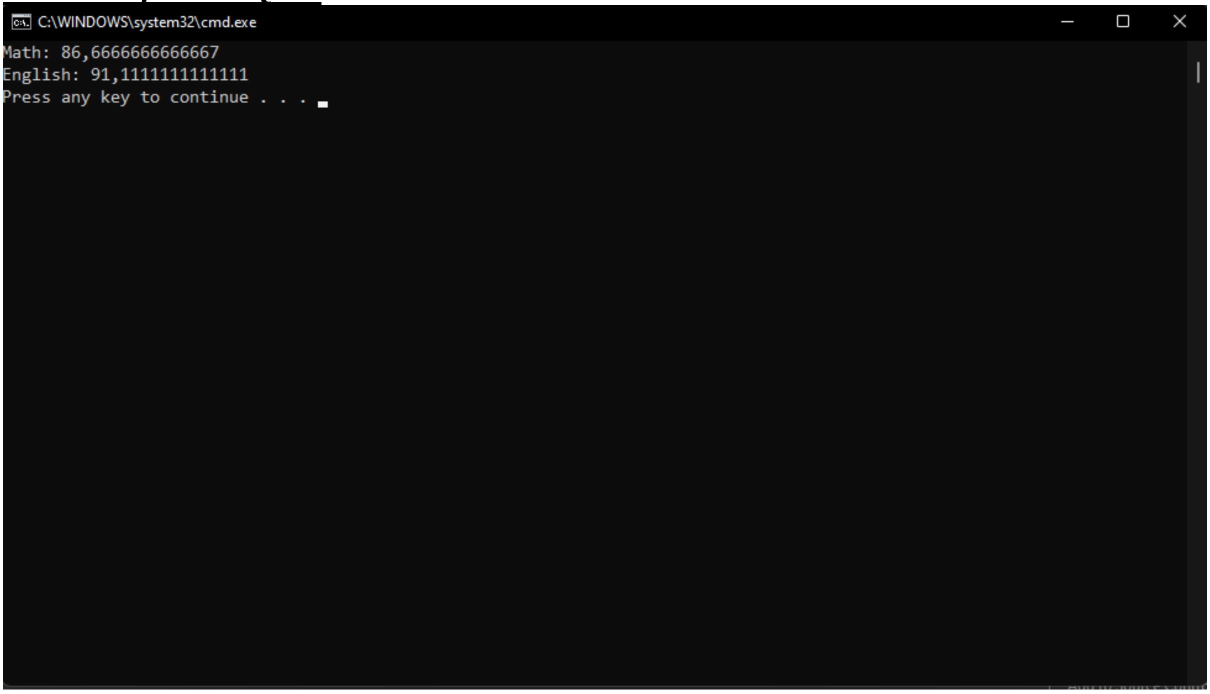
Question 2.2

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.

The code for Q 2.2

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

namespace student
{
   public class Student
   {
      public string Name { get; set; }
      public int Age { get; set; }
      public int[] Grades { get; set; }
   }

   public class Course
   {
      public string Name { get; set; }
      public List<Student> Students { get; set; }
   }

   class Program
   {
      static void Main(string[] args)
      {
         List<Course> courses = new List<Course>
         {
            new Course
            {
               Name = "Math",
               Students = new List<Student>
               {
                  new Student { Name = "Alice", Age = 22, Grades = new int[] {
80, 85, 90 } },
                        new Student { Name = "Bob", Age = 21, Grades = new int[] {
75, 80, 85 } },
                        new Student { Name = "Charlie", Age = 23, Grades = new int[] {
90, 95, 100 } }
                    }
                },
                new Course
                {
                    Name = "English",
                    Students = new List<Student>
                    {
                        new Student { Name = "David", Age = 20, Grades = new int[] {
85, 90, 95 } },
                        new Student { Name = "Emma", Age = 22, Grades = new int[] {
95, 100, 100 } },
                        new Student { Name = "Frank", Age = 21, Grades = new int[] {
80, 85, 90 } }
                    }
                }
             };
            var query = from course in courses
                        select new
                        {
                            CourseName = course.Name,
                            AverageGrade = course.Students.SelectMany(s =>
s.Grades).Average()
                        };

            // execute the query and print the results
            foreach (var result in query)
            {
                Console.WriteLine($"{result.CourseName}:
{result.AverageGrade}");
            }
        }
    }
}
```

Code output for Q2.2



```
Math: 86,6666666666667
English: 91,1111111111111
Press any key to continue . . .
```

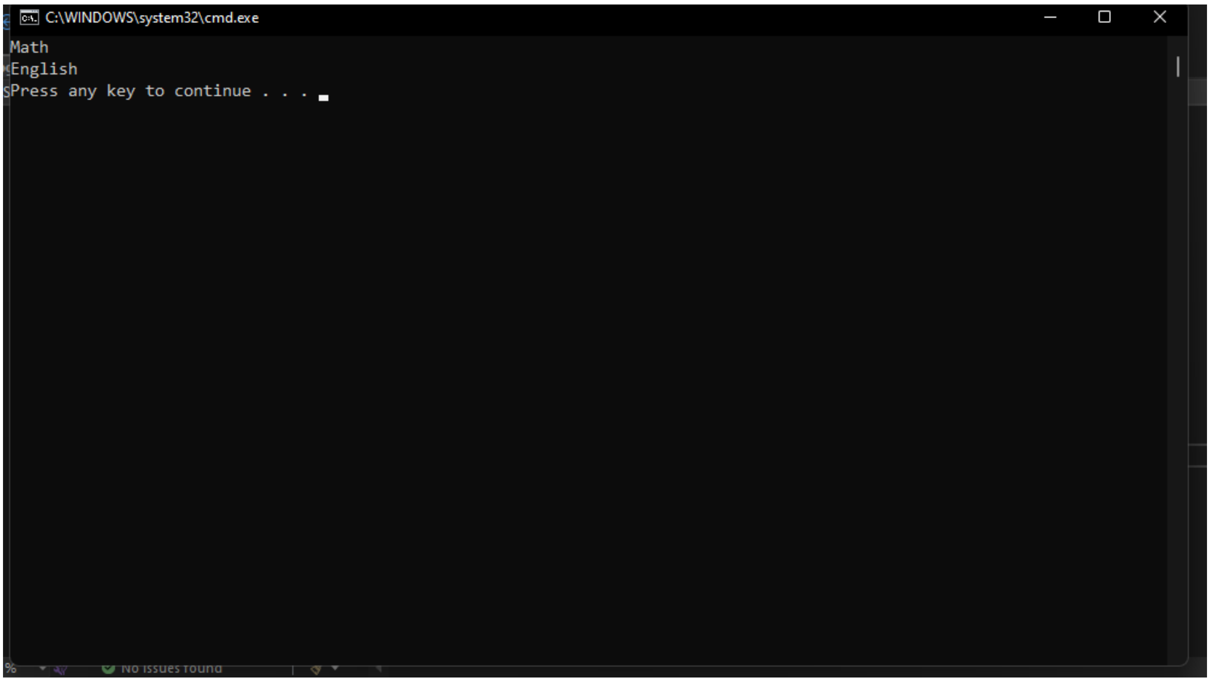
Question 2.3

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.

The code with Q 2.3

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

namespace student
{
   public class Student
   {
      public string Name { get; set; }
      public int Age { get; set; }
      public int[] Grades { get; set; }
   }

   public class Course
   {
      public string Name { get; set; }
      public List<Student> Students { get; set; }
   }

   class Program
   {
      static void Main(string[] args)
      {
         List<Course> courses = new List<Course>
         {
            new Course
            {
               Name = "Math",
               Students = new List<Student>
               {
                  new Student { Name = "Alice", Age = 22, Grades = new int[] {
80, 85, 90 } },
                  new Student { Name = "Bob", Age = 21, Grades = new int[] {
75, 80, 85 } },
                        new Student { Name = "Charlie", Age = 23, Grades = new int[] { 90, 95, 100 } }
                    }
                },
                new Course
                {
                    Name = "English",
                    Students = new List<Student>
                    {
                        new Student { Name = "David", Age = 20, Grades = new int[] { 85, 90, 95 } },
                        new Student { Name = "Emma", Age = 22, Grades = new int[] { 95, 100, 100 } },
                        new Student { Name = "Frank", Age = 21, Grades = new int[] { 80, 85, 90 } }
                    }
                }
             };
            var query = from course in courses
                        where course.Students.All(s => s.Grades.Any(g => g >= 80))
                        select course.Name;

            // execute the query and print the results
            foreach (var result in query)
            {
                Console.WriteLine(result);
            }
        }
    }
}
```

The output for Q 2.3



```
Math
English
Press any key to continue . . .
```

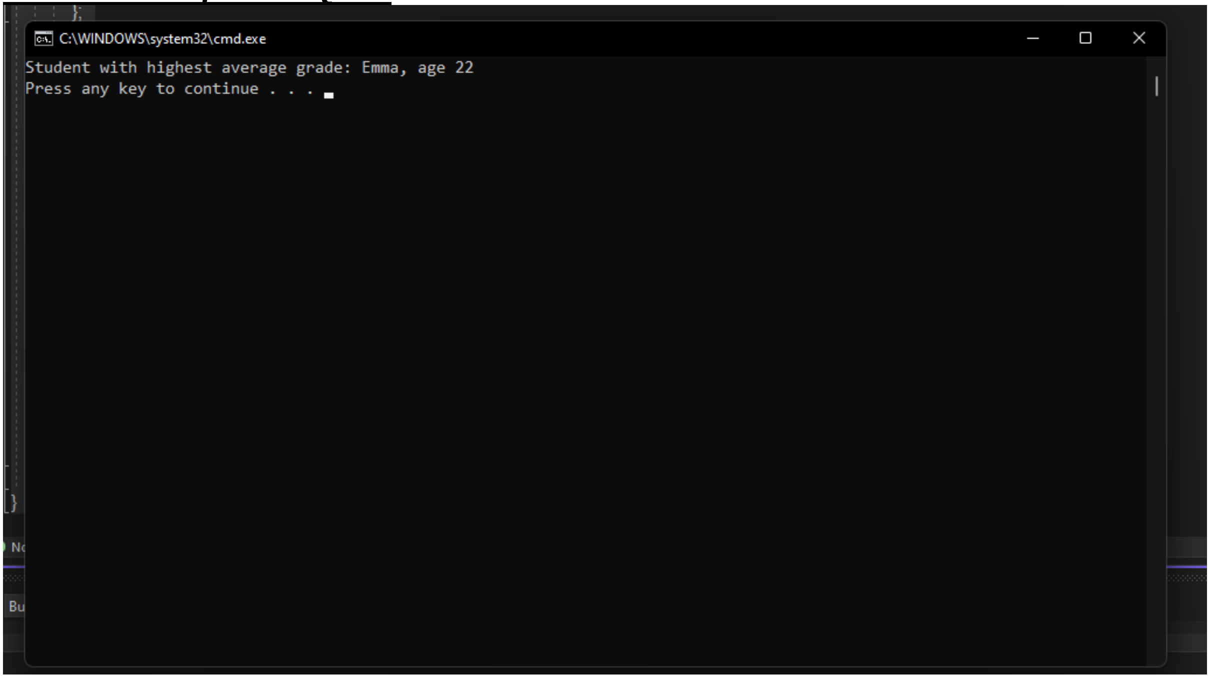
Question 2 .4

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

The code for Q 2.4

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

namespace student
{
   public class Student
   {
      public string Name { get; set; }
      public int Age { get; set; }
      public int[] Grades { get; set; }
   }

   public class Course
   {
      public string Name { get; set; }
      public List<Student> Students { get; set; }
   }

   class Program
   {
      static void Main(string[] args)
      {
         List<Course> courses = new List<Course>
         {
            new Course
            {
               Name = "Math",
               Students = new List<Student>
               {
                  new Student { Name = "Alice", Age = 22, Grades = new int[] {
80, 85, 90 } },
                  new Student { Name = "Bob", Age = 21, Grades = new int[] {
75, 80, 85 } },
                        new Student { Name = "Charlie", Age = 23, Grades = new int[] {
90, 95, 100 } }
                    }
                },
                new Course
                {
                    Name = "English",
                    Students = new List<Student>
                    {
                        new Student { Name = "David", Age = 20, Grades = new int[] {
85, 90, 95 } },
                        new Student { Name = "Emma", Age = 22, Grades = new int[] {
95, 100, 100 } },
                        new Student { Name = "Frank", Age = 21, Grades = new int[] {
80, 85, 90 } }
                    }
                }
             };
            var query = from course in courses
                        from student in course.Students
                        select new
                        {
                            Name = student.Name,
                            Age = student.Age,
                            AverageGrade = student.Grades.Average()
                        } into studentGrades
                        orderby studentGrades.AverageGrade descending
                        select new
                        {
                            Name = studentGrades.Name,
                            Age = studentGrades.Age
                        };

            // execute the query and print the result
            var result = query.First();
            Console.WriteLine($"Student with highest average grade:
{result.Name}, age {result.Age}");
        }
    }
}
```

The code output for Q 2.4



```
Student with highest average grade: Emma, age 22
Press any key to continue . . .
```

Declaration

Completed Declaration of Authenticity

I ____Michaeleah Anne Murray______ hereby declare that the contents of this assignment is entirely my own work except for the following documents: (List the documents and page numbers of work in this portfolio
that were generated in a group)

Activity	Date
PRG521 FA2	23 April 2023

Signature:_ M.A.K.M__________ Date: ___23 April 2023_________________