Object Oriented Programming
Lab Manual 1

Introduction
In this course, we will learn a new programming language that is known as C# (C Sharp). Now that you have successfully installed and set up the Visual Studio on your computer and learned everything till Arrays, let's now learn the remaining concepts of functions and file handling.

Let's do some coding.

Functions in C#
We have learned in the previous manual about various operations in C#. In this section, we will implement the functions.

Syntax:

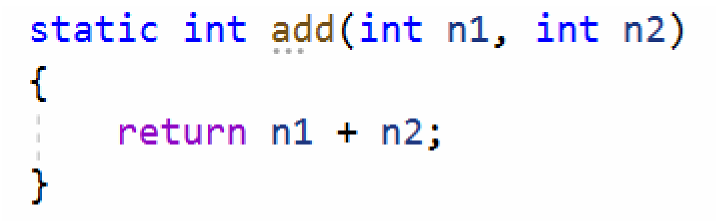
```
static return_type function_name (comma separated arguments)
{
        // statements
        // return statement
}
```

Task: To understand this concept, try implementing the following program. Write a Function named addition that takes two parameters as input and then returns their sum.

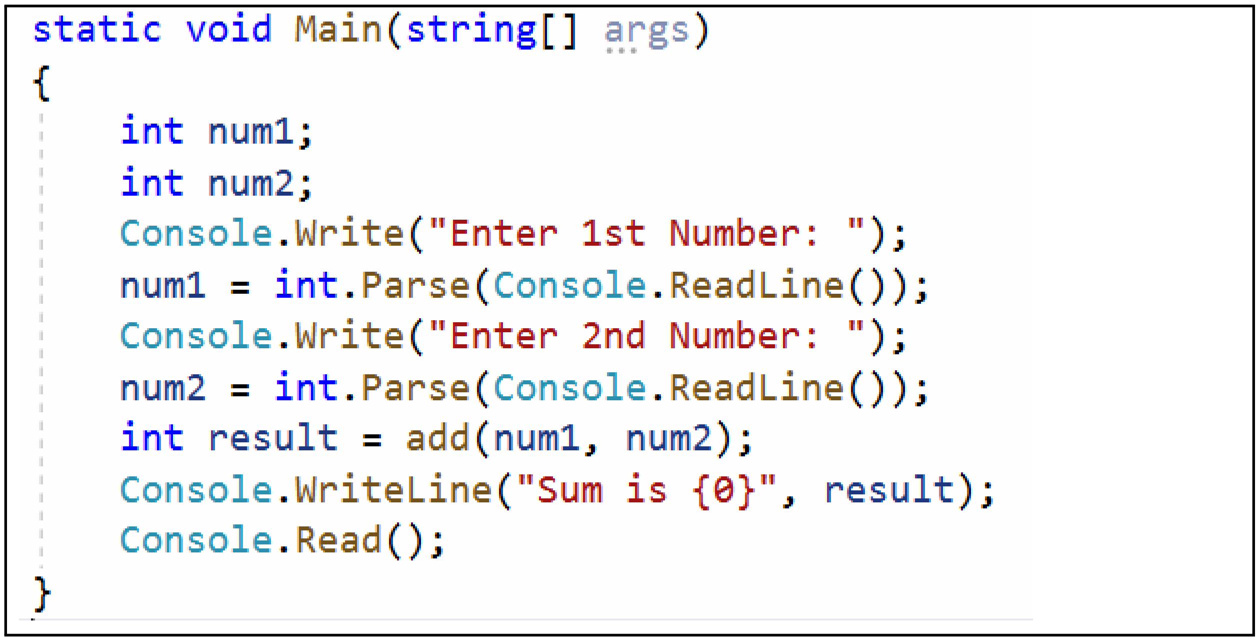
Solution:

Write the following code into the main function of the code and execute the program by clicking on the start button.

Code:



```
static int add(int n1, int n2)
{
    return n1 + n2;
}
```



```
static void Main(string[] args)
{
    int num1;
    int num2;
    Console.Write("Enter 1st Number: ");
    num1 = int.Parse(Console.ReadLine());
    Console.Write("Enter 2nd Number: ");
    num2 = int.Parse(Console.ReadLine());
    int result = add(num1, num2);
    Console.WriteLine("Sum is {0}", result);
    Console.Read();
}
```

```
static int add(int n1, int n2)
{
        return n1+n2;
}
```

This function will calculate the sum of both numbers provided as arguments to it and return that sum to the main function. The rest of the functionality of the program has been explained in detail in the previous lab manuals.

Ma Sha ALLAH Students, You are en route to becoming a kamyab programmer.
Let's Continue the fun coding Ride.

File Handling in C#

File Handling is, without any doubt, the most important feature of any programming language. C# also provides the functionality and support for file handling, however, there is a small difference of syntax here. Look at the below code to understand the difference and implementation.

Reading from a File

The syntax of the **file read** is mentioned below.

Syntax:

File.Exists("path");
// returns true if a readable file exists at the given path

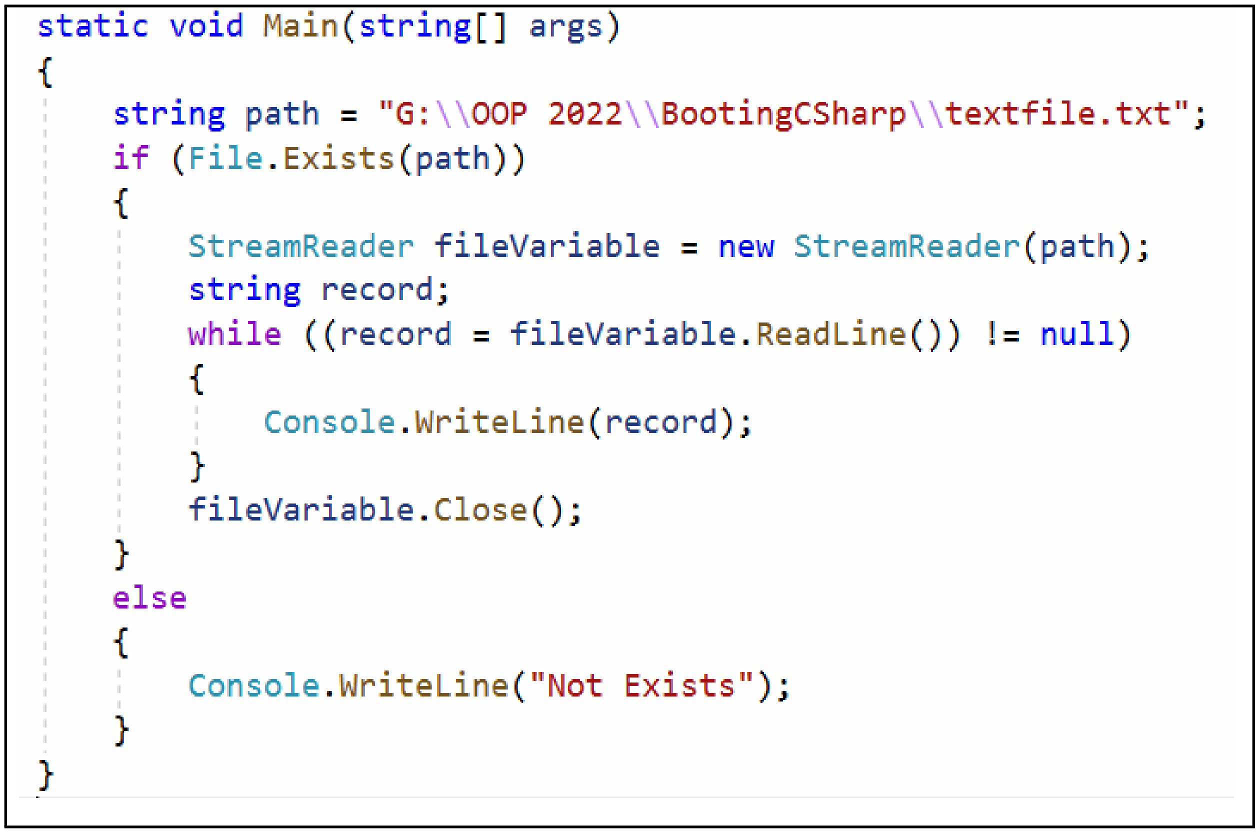
StreamReader file_variable_name = new StreamReader(path);
// Creates a file handling variable

file_variable_name.ReadLine();
// reads and returns a complete line from the file

using System.IO;
// import the necessary files for using StreamReader type variables

Task: To understand this concept, try writing a program that reads the data line by line from the file after checking if the file exists.
Solution:
Write the following code into the main function of the code and execute the program by clicking on the start button.

Code:



```
static void Main(string[] args)
{
    string path = "G:\\OOP 2022\\BootingCSharp\\textfile.txt";
    if (File.Exists(path))
    {
        StreamReader fileVariable = new StreamReader(path);
        string record;
        while ((record = fileVariable.ReadLine()) != null)
        {
            Console.WriteLine(record);
        }
        fileVariable.Close();
    }
    else
    {
        Console.WriteLine("Not Exists");
    }
}
```

This program checks if there exists a file at the given path and then returns the contents of the file line by line from the file.

Writing to a File

The syntax of the **file write** is mentioned below.

Syntax:

StreamWriter file_variable_name = new StreamWriter(path, true);
// Creates file handling variable for Writing data to variable

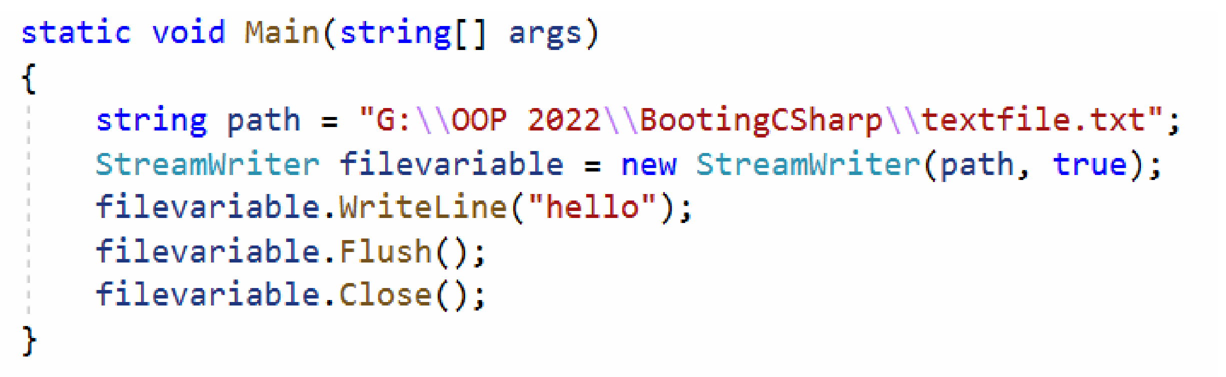
file_variable_name.WriteLine("Write your message here");
// Used to write data into the file

file_variable_name.Flush();
// Used to empty the stream buffer by writing all the remaining data into the file

Task: To understand this concept, try writing a program that appends the data into the file.

Solution:
Write the following code into the main function of the code and execute the program by clicking on the start button.

Code:



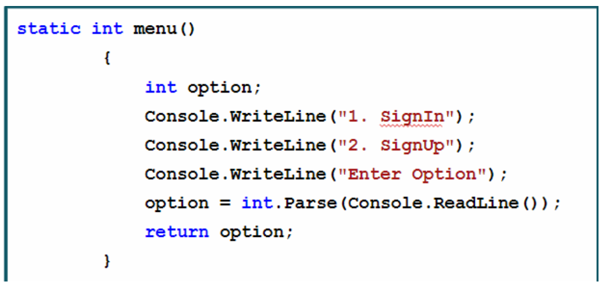
```
static void Main(string[] args)
{
    string path = "G:\\OOP 2022\\BootingCSharp\\textfile.txt";
    StreamWriter filevariable = new StreamWriter(path, true);
    filevariable.WriteLine("hello");
    filevariable.Flush();
    filevariable.Close();
}
```

This program will write the “hello” string on the file stored at the defined path location. Moreover, it is mandatory to always close the file after you have performed your required functionality.

Sign In and Sign Up Application

Make a SignIn and SignUp Application that will check if the user is stored in the file then it will allow it to log in. If the user SignUp then the record is stored in the file in comma-separated format.

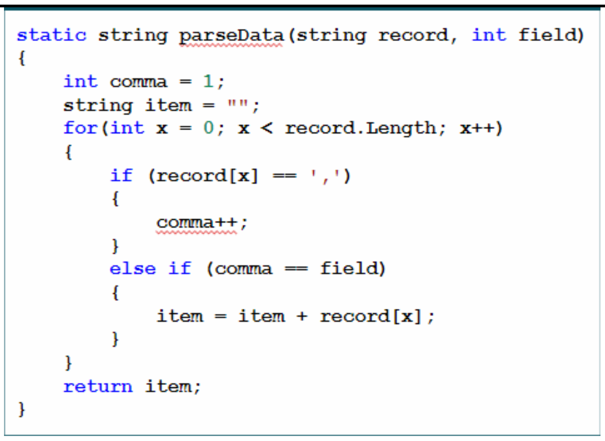
Step1:
Creating the Menu



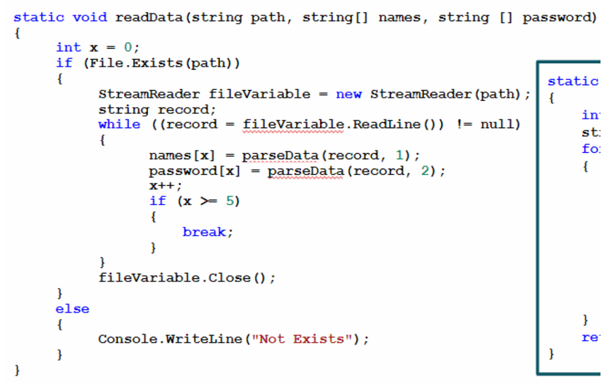
```
static int menu()
        {
            int option;
            Console.WriteLine("1. SignIn");
            Console.WriteLine("2. SignUp");
            Console.WriteLine("Enter Option");
            option = int.Parse(Console.ReadLine());
            return option;
        }
```

Step 2:

Reading data from file

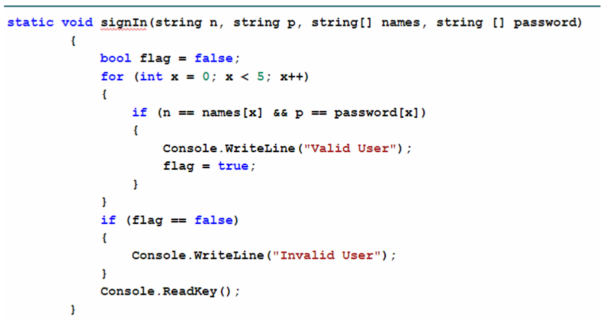


```
static string parseData(string record, int field)
{
    int comma = 1;
    string item = "";
    for(int x = 0; x < record.Length; x++)
    {
        if (record[x] == ',')
        {
            comma++;
        }
        else if (comma == field)
        {
            item = item + record[x];
        }
    }
    return item;
}
```



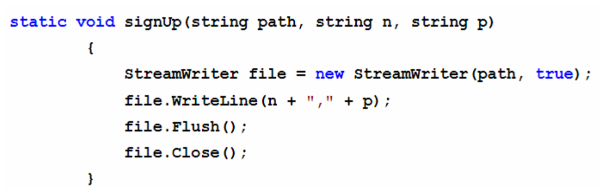
```
static void readData(string path, string[] names, string [] password)
{
    int x = 0;
    if (File.Exists(path))
    {
        StreamReader fileVariable = new StreamReader(path);
        string record;
        while ((record = fileVariable.ReadLine()) != null)
        {
            names[x] = parseData(record, 1);
            password[x] = parseData(record, 2);
            x++;
            if (x >= 5)
            {
                break;
            }
        }
        fileVariable.Close();
    }
    else
    {
        Console.WriteLine("Not Exists");
    }
}
```

Step 3:
Make the SignIn Function



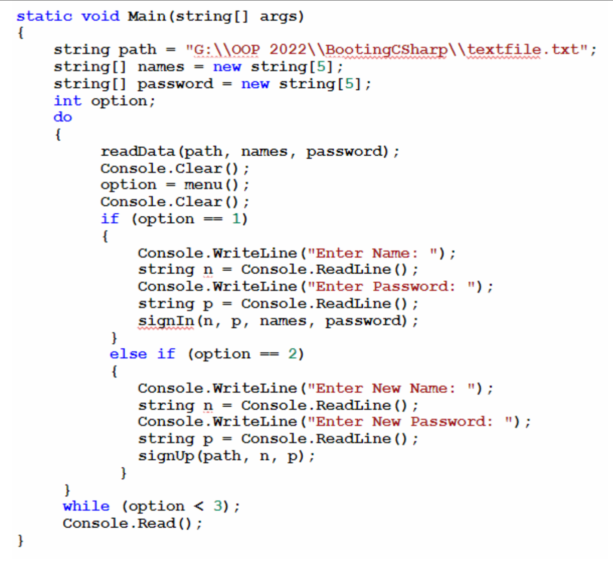
```
static void signIn(string n, string p, string[] names, string [] password)
        {
            bool flag = false;
            for (int x = 0; x < 5; x++)
            {
                if (n == names[x] && p == password[x])
                {
                    Console.WriteLine("Valid User");
                    flag = true;
                }
            }
            if (flag == false)
            {
                Console.WriteLine("Invalid User");
            }
            Console.ReadKey();
        }
```

Step 4:
Make the SignUp Function



```
static void signUp(string path, string n, string p)
        {
            StreamWriter file = new StreamWriter(path, true);
            file.WriteLine(n + "," + p);
            file.Flush();
            file.Close();
        }
```

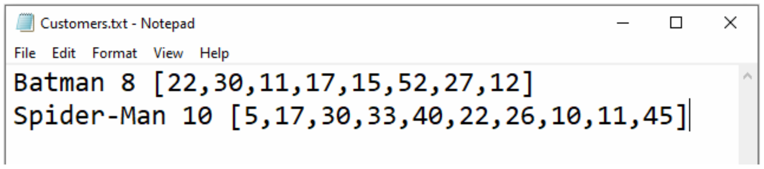
Step 5:
Main Function



```
static void Main(string[] args)
{
    string path = "G:\\OOP 2022\\BootingCSharp\\textfile.txt";
    string[] names = new string[5];
    string[] password = new string[5];
    int option;
    do
    {
         readData(path, names, password);
         Console.Clear();
         option = menu();
         Console.Clear();
         if (option == 1)
         {
             Console.WriteLine("Enter Name: ");
             string n = Console.ReadLine();
             Console.WriteLine("Enter Password: ");
             string p = Console.ReadLine();
             signIn(n, p, names, password);
          }
          else if (option == 2)
          {
             Console.WriteLine("Enter New Name: ");
             string n = Console.ReadLine();
             Console.WriteLine("Enter New Password: ");
             string p = Console.ReadLine();
             signUp(path, n, p);
           }
     }
     while (option < 3);
     Console.Read();
}
```

SELF ASSESSMENT TASKS

KamyabLife is launching a network of autonomous pizza delivery drones and wants you to create a flexible rewards system (Pizza Points) that can be tweaked in the future. The rules are simple:
if a customer has made at least N orders of at least Y price, they get a FREE pizza!
The information of the customers is stored in a file Customers.txt in the following format. First, the name of the customer is given then after the space the number of orders is given then after the space within the brackets all the orders prices are given.



```
Batman 8 [22,30,11,17,15,52,27,12]
Spider-Man 10 [5,17,30,33,40,22,26,10,11,45]
```

Your task is to create a function that takes a minimum number of orders and a minimum order price then displays the names of the customers that are eligible for a free pizza.

Test Cases

Input	Output
pizza_points(5, 20)	"Spider-Man"

pizza_points(3, 10)	"Batman" "Spider-Man"
pizza_points(5, 100)	“”

Good Luck and Best Wishes !!

Happy Coding ahead :)