**University of Kotli Azad Jammu & Kashmir**

**Faculty of Computing & Engineering**

**Department Software Engineering**

**A blue circle with a logo and text

Description automatically generatedAssignment # 01**

**Topic:** ISPs

**Course:** Software Quality Assurance

**Submitted By:**

|  |  |
| --- | --- |
| **Name** | **Roll No** |
| **Qura-Tul-Ain Saleem** | **04** |

**Class: BS SE**

**Semester: 6th**

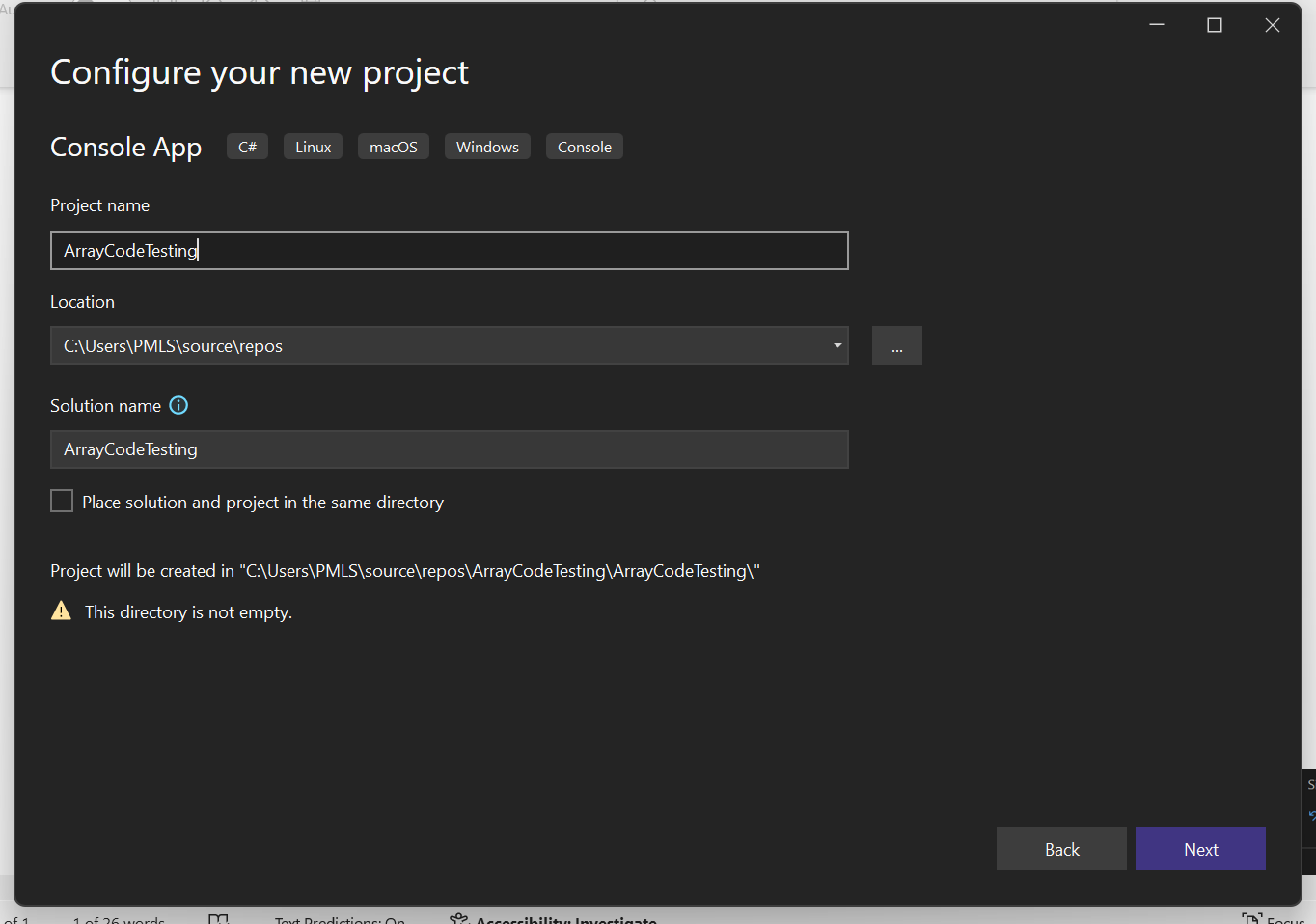
**Submitted to: Sir Khurrum**

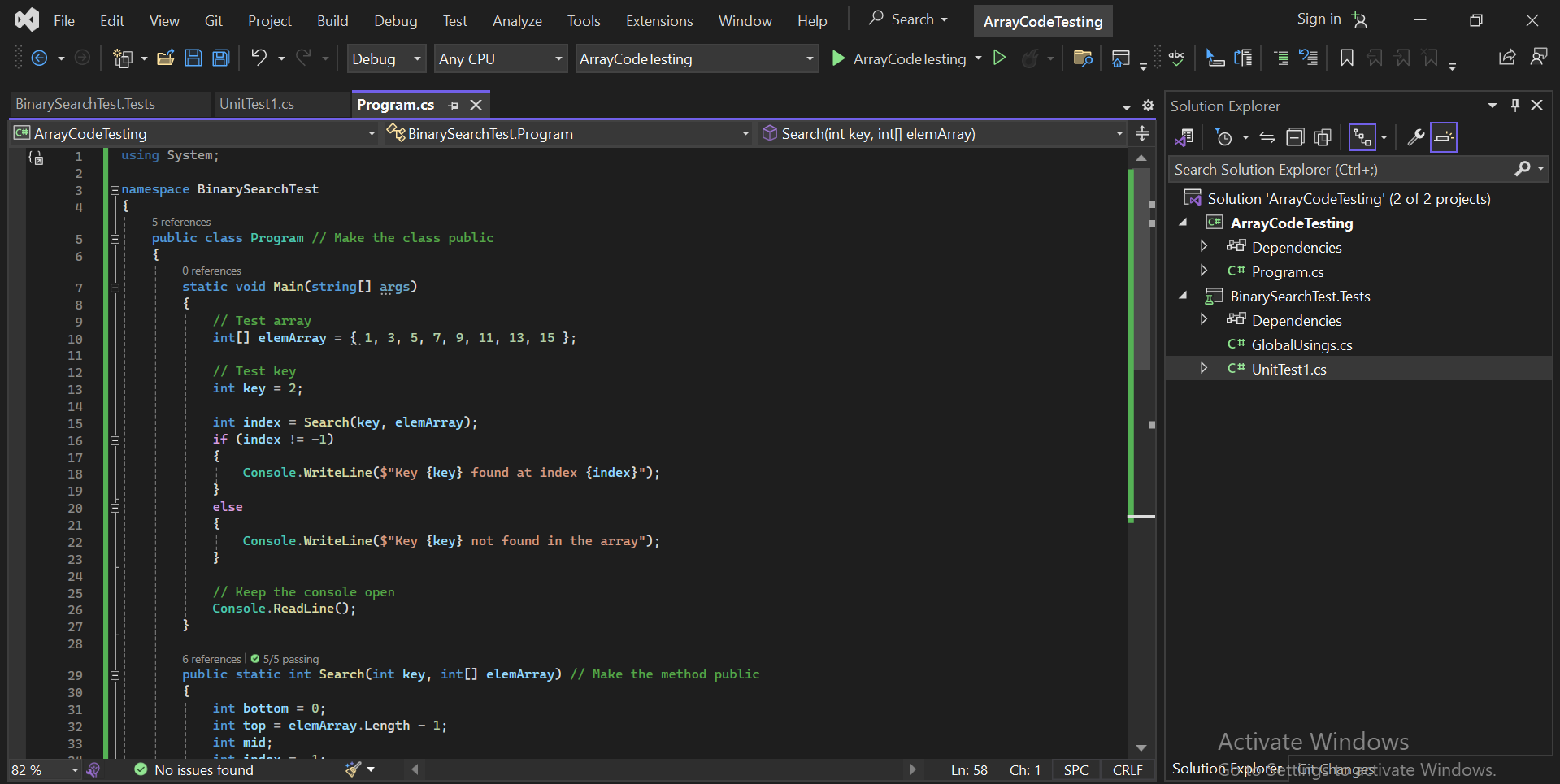
**Submission Date:** 10-10-2024

**Step 1: Set Up the projects**

1. **Create a New Project:**
   * Open Visual Studio and create a new solution called ArrayCodeTesting.
2. **Add a Console Application Project**:

* Add a new project to the solution and choose **Console App**.
* Name the project ArrayCodeTesting.
* Write the implementation of the Search method in the Program.cs file.





**Code :**

using System;

namespace BinarySearchTest

{

public class Program // Make the class public

{

static void Main(string[] args)

{

// Test array

int[] elemArray = { 1, 3, 5, 7, 9, 11, 13, 15 };

// Test key

int key = 2;

int index = Search(key, elemArray);

if (index != -1)

{

Console.WriteLine($"Key {key} found at index {index}");

}

else

{

Console.WriteLine($"Key {key} not found in the array");

}

// Keep the console open

Console.ReadLine();

}

public static int Search(int key, int[] elemArray) // Make the method public

{

int bottom = 0;

int top = elemArray.Length - 1;

int mid;

int index = -1;

bool found = false;

while (bottom <= top && !found)

{

mid = (top + bottom) / 2;

if (elemArray[mid] == key)

{

index = mid;

found = true;

return index;

}

else

{

if (elemArray[mid] < key)

bottom = mid + 1;

else

top = mid - 1;

}

}

return index;

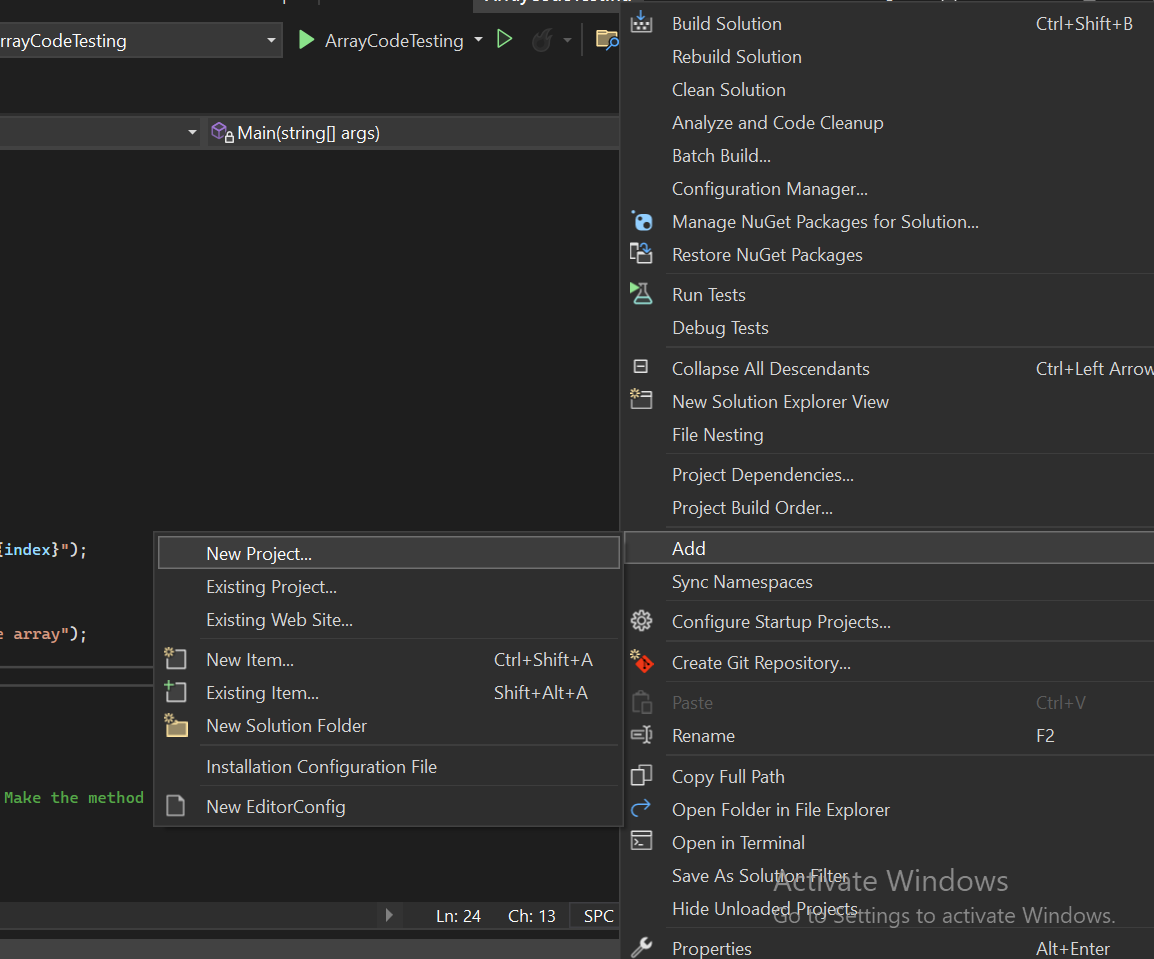
}

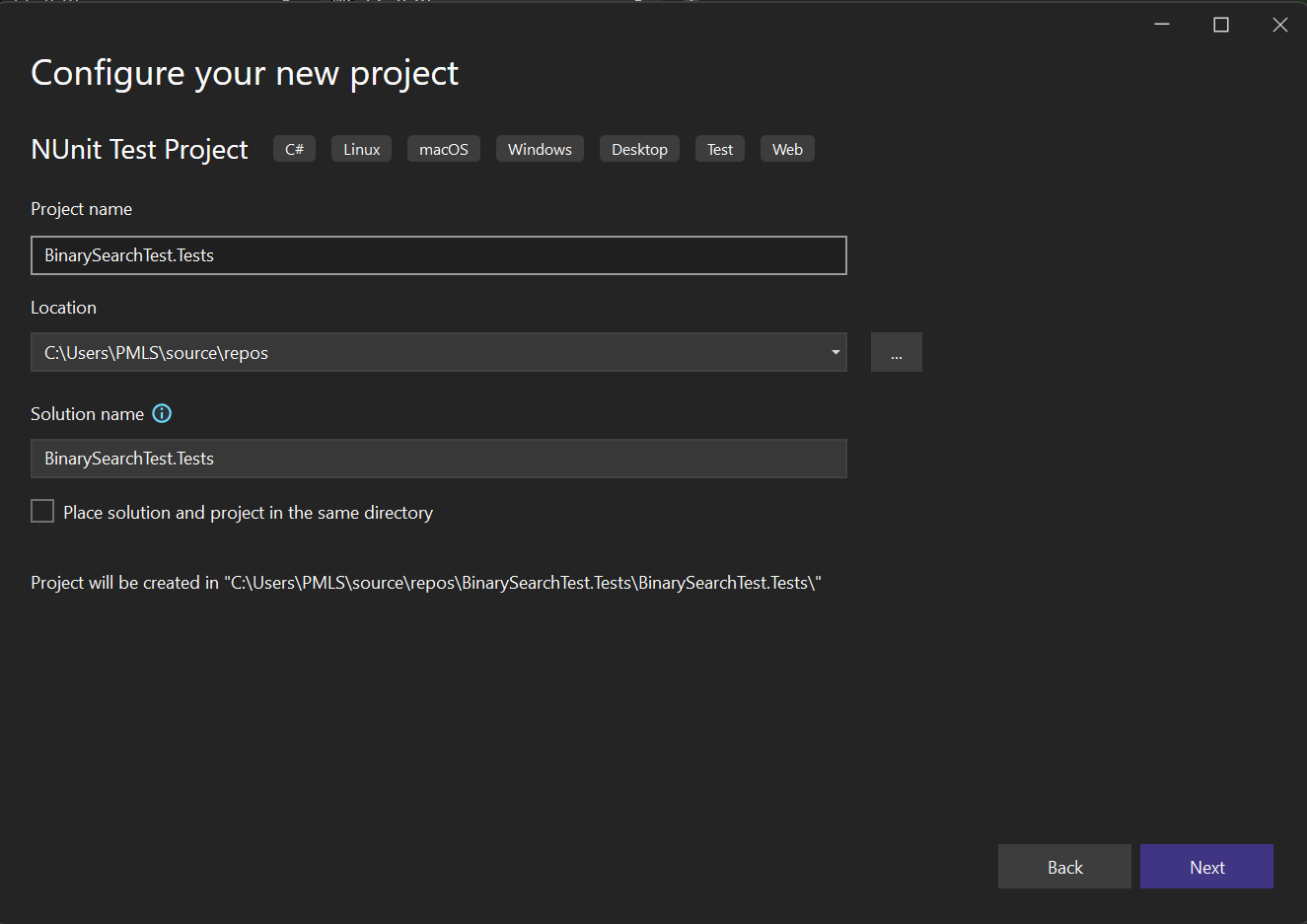
}

}

1. **Add a Unit Test Project**:

* Right-click on the solution and select **Add** > **New Project...**.
* Choose **NUnit Test Project (.NET Core)** (if not available, you can choose **xUnit Test Project** or **MSTest Test Project**).
* Name the test project BinarySearchTest.Tests.





1. **Add a Reference to the Main Project**:

* Right-click on the BinarySearchTest.Tests project in **Solution Explorer**.
* Select **Add** > **Project Reference...**.In the **Reference Manager**, check the box for ArrayCodeTesting and click **OK**.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

**Step 4: Run the Tests in Visual Studio**

1. **Open Test Explorer**:
   * Go to the **Test** menu and select **Test Explorer** to open the Test Explorer window where all tests will be displayed.
2. **Run the Tests**:
   * Click **Run All** in the Test Explorer to run all the tests in the BinarySearchTest.Tests project.
   * Alternatively, you can run individual tests by right-clicking on a test name and selecting **Run**.

Code :

using NUnit.Framework; // Import NUnit for unit testing

using BinarySearchTest; // Reference your main project namespace

namespace BinarySearchTest.Tests

{

[TestFixture] // Indicates that this class contains NUnit tests

internal class ProgramTests

{

[Test] // Marks this method as a test

public void Search\_KeyExists\_ReturnsCorrectIndex()

{

// Arrange

int[] elemArray = { 1, 3, 5, 7, 9, 11, 13, 15 };

int key = 5;

int expectedIndex = 2;

// Act

int result = Program.Search(key, elemArray);

// Assert

Assert.AreEqual(expectedIndex, result);

}

[Test]

public void Search\_KeyDoesNotExist\_ReturnsNegativeOne()

{

// Arrange

int[] elemArray = { 1, 3, 5, 7, 9, 11, 13, 15 };

int key = 2;

// Act

int result = Program.Search(key, elemArray);

// Assert

Assert.AreEqual(-1, result);

}

[Test]

public void Search\_EmptyArray\_ReturnsNegativeOne()

{

// Arrange

int[] elemArray = { };

int key = 5;

// Act

int result = Program.Search(key, elemArray);

// Assert

Assert.AreEqual(-1, result);

}

[Test]

public void Search\_KeyAtStart\_ReturnsZero()

{

// Arrange

int[] elemArray = { 1, 3, 5, 7, 9, 11, 13, 15 };

int key = 1;

int expectedIndex = 0;

// Act

int result = Program.Search(key, elemArray);

// Assert

Assert.AreEqual(expectedIndex, result);

}

[Test]

public void Search\_KeyAtEnd\_ReturnsLastIndex()

{

// Arrange

int[] elemArray = { 1, 3, 5, 7, 9, 11, 13, 15 };

int key = 15;

int expectedIndex = elemArray.Length - 1;

// Act

int result = Program.Search(key, elemArray);

// Assert

Assert.AreEqual(expectedIndex, result);

}

}

}

A screenshot of a computer program

Description automatically generated