**SOFTWARE ENGENEERING**

**5 BCA B**

**"Practical - 4"**

***BY***

**"sarthak" (23215134)**

**SUBMITTED TO**

**kuljeet singh**

****

**SCHOOL OF SCIENCES**

**2025-2026**

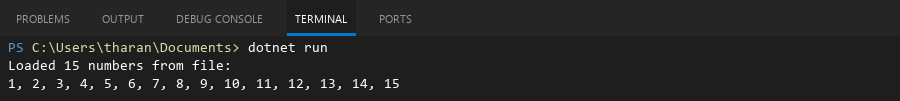
## **QUESTION 1**

***1. Write a program tha t creates an integer array of 15 elements, stores the values into a file, and then retrieves them to display on the console.***

### **Code Solution**

using System;  
using System.IO;  
  
class Program  
{  
 static void Main()  
 {  
 int[] numbers = new int[15] { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 };  
   
 File.WriteAllText("data.txt", string.Join(",", numbers));  
   
 string content = File.ReadAllText("data.txt");  
 string[] stringNumbers = content.Split(',');  
 int[] loadedNumbers = new int[stringNumbers.Length];  
   
 for (int i = 0; i < stringNumbers.Length; i++)  
 {  
 loadedNumbers[i] = int.Parse(stringNumbers[i]);  
 }  
   
 Console.WriteLine("Loaded " + loadedNumbers.Length + " numbers from file:");  
 Console.WriteLine(string.Join(", ", loadedNumbers));  
 }  
}

### **FINAL Output**



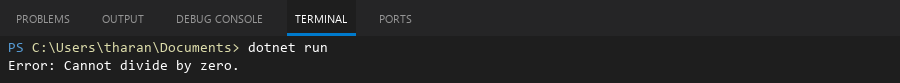
## **QUESTION 2**

***2. Write a program to input two integers and divide them. Use a try -catch block to handle the DivideByZeroException and display an appropriate message. Further, if the data type of the elements do not match with defined type then throw an exception too.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int numerator = 10;  
 int denominator = 0;  
   
 try  
 {  
 int result = numerator / denominator;  
 Console.WriteLine("Result: " + result);  
 }  
 catch (DivideByZeroException)  
 {  
 Console.WriteLine("Error: Cannot divide by zero.");  
 }  
 catch (Exception)  
 {  
 Console.WriteLine("Error: Data type mismatch.");  
 }  
 }  
}

### **FINAL Output**



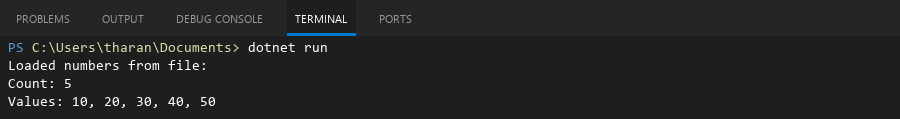
## **QUESTION 3**

***3. Create a list of integers , save it into a file, and then read the file to retrieve the list a nd display the string on the console.***

### **Code Solution**

using System;  
using System.Collections.Generic;  
using System.IO;  
  
class Program  
{  
 static void Main()  
 {  
 List<int> numbers = new List<int> { 10, 20, 30, 40, 50 };  
 string filePath = "data.txt";  
  
 using (StreamWriter writer = new StreamWriter(filePath))  
 {  
 foreach (int num in numbers)  
 {  
 writer.WriteLine(num);  
 }  
 }  
  
 List<int> loadedNumbers = new List<int>();  
 using (StreamReader reader = new StreamReader(filePath))  
 {  
 string line;  
 while ((line = reader.ReadLine()) != null)  
 {  
 if (int.TryParse(line, out int value))  
 {  
 loadedNumbers.Add(value);  
 }  
 }  
 }  
  
 Console.WriteLine("Loaded numbers from file:");  
 Console.WriteLine("Count: " + loadedNumbers.Count);  
 Console.WriteLine("Values: " + string.Join(", ", loadedNumbers));  
 }  
}

### **FINAL Output**



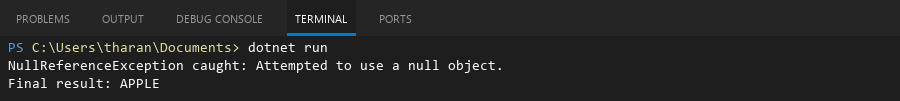
## **QUESTION 4**

***4. Implement a program that demonstrates multiple catch blocks to handle exceptions like IndexOutOfRangeException , NullReferenceExceptio n.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 object[] items = { "apple", null, "cherry" };  
 string result = "";  
  
 try  
 {  
 for (int i = 0; i <= items.Length; i++)  
 {  
 string item = (string)items[i];  
 result += item.ToUpper() + " ";  
 }  
 }  
 catch (IndexOutOfRangeException)  
 {  
 Console.WriteLine("IndexOutOfRangeException caught: Attempted to access beyond array bounds.");  
 }  
 catch (NullReferenceException)  
 {  
 Console.WriteLine("NullReferenceException caught: Attempted to use a null object.");  
 }  
 finally  
 {  
 Console.WriteLine("Final result: " + result);  
 }  
 }  
}

### **FINAL Output**



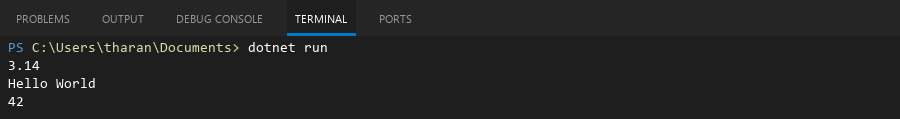
## **QUESTION 5**

***5. Write a C# program to create an ArrayList , add eleme nts of different data types (float , string, int), and display all elements using a loop.***

### **Code Solution**

using System;  
using System.Collections;  
  
class Program  
{  
 static void Main()  
 {  
 ArrayList myList = new ArrayList();  
 myList.Add(3.14f);  
 myList.Add("Hello World");  
 myList.Add(42);  
  
 foreach (var element in myList)  
 {  
 Console.WriteLine(element);  
 }  
 }  
}

### **FINAL Output**



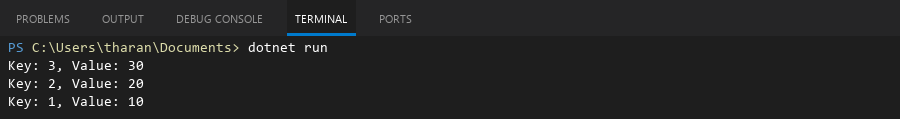
## **QUESTION 6**

***6. Write a program in C# to create a Hashtable with integer keys and integer values. Insert three key -value pairs and display them using a loop.***

### **Code Solution**

using System;  
using System.Collections;  
  
class Program  
{  
 static void Main()  
 {  
 Hashtable hashtable = new Hashtable();  
 hashtable.Add(1, 10);  
 hashtable.Add(2, 20);  
 hashtable.Add(3, 30);  
  
 foreach (DictionaryEntry entry in hashtable)  
 {  
 Console.WriteLine("Key: " + entry.Key + ", Value: " + entry.Value);  
 }  
 }  
}

### **FINAL Output**



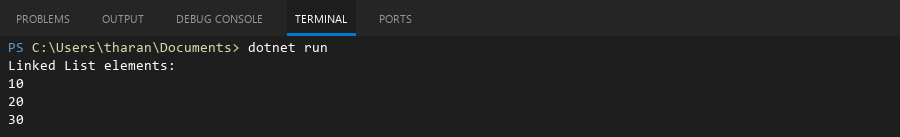
## **QUESTION 7**

***7. Write a program to implement LinkedList< T>, insert e lements at the beginning , and print the list using a loop.***

### **Code Solution**

using System;  
using System.Collections.Generic;  
  
class Program  
{  
 static void Main()  
 {  
 LinkedList<int> list = new LinkedList<int>();  
   
 list.AddFirst(30);  
 list.AddFirst(20);  
 list.AddFirst(10);  
   
 Console.WriteLine("Linked List elements:");  
 foreach (int item in list)  
 {  
 Console.WriteLine(item);  
 }  
 }  
}

### **FINAL Output**



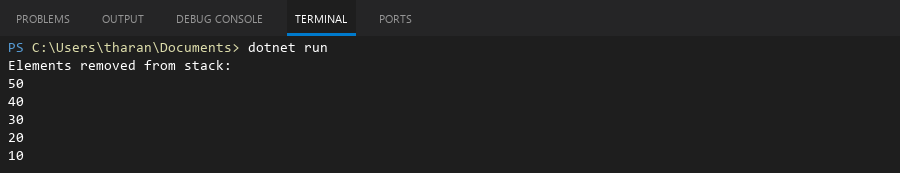
## **QUESTION 8**

***8. Write a program to implement Stack , insert five elements and remove them .***

### **Code Solution**

using System;  
using System.Collections.Generic;  
  
class Program  
{  
 static void Main()  
 {  
 Stack<int> stack = new Stack<int>();  
   
 stack.Push(10);  
 stack.Push(20);  
 stack.Push(30);  
 stack.Push(40);  
 stack.Push(50);  
   
 Console.WriteLine("Elements removed from stack:");  
   
 while (stack.Count > 0)  
 {  
 int element = stack.Pop();  
 Console.WriteLine(element);  
 }  
 }  
}

### **FINAL Output**



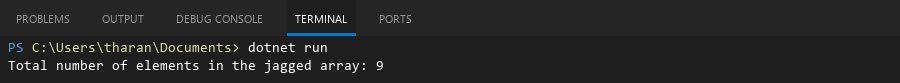
## **QUESTION 9**

***9. Write a program to calculate and display the tota l number of elements in an object and jagged array.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[][] jaggedArray = new int[][]  
 {  
 new int[] { 1, 2, 3 },  
 new int[] { 4, 5 },  
 new int[] { 6, 7, 8, 9 }  
 };  
  
 int totalElements = 0;  
 for (int i = 0; i < jaggedArray.Length; i++)  
 {  
 totalElements += jaggedArray[i].Length;  
 }  
  
 Console.WriteLine("Total number of elements in the jagged array: " + totalElements);  
 }  
}

### **FINAL Output**



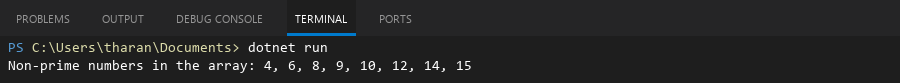
## **QUESTION 10**

***10. Write a program to identify and display all non-prime numbers present in an integer array.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[] numbers = { 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 };  
 bool[] isPrime = new bool[numbers.Length];  
   
 for (int i = 0; i < numbers.Length; i++)  
 {  
 if (numbers[i] < 2)  
 {  
 isPrime[i] = false;  
 continue;  
 }  
   
 isPrime[i] = true;  
 for (int j = 2; j \* j <= numbers[i]; j++)  
 {  
 if (numbers[i] % j == 0)  
 {  
 isPrime[i] = false;  
 break;  
 }  
 }  
 }  
   
 Console.Write("Non-prime numbers in the array: ");  
 bool first = true;  
 for (int i = 0; i < numbers.Length; i++)  
 {  
 if (!isPrime[i])  
 {  
 if (!first)  
 {  
 Console.Write(", ");  
 }  
 Console.Write(numbers[i]);  
 first = false;  
 }  
 }  
 Console.WriteLine();  
 }  
}

### **FINAL Output**



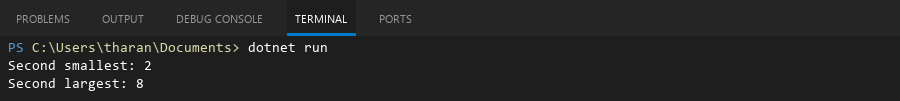
## **QUESTION 11**

***11. Write a program to find and displ ay the second largest and smallest numbers in an array.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[] numbers = { 5, 2, 8, 1, 9, 3, 7, 4, 6 };  
   
 if (numbers.Length < 2)  
 {  
 Console.WriteLine("Array must contain at least 2 elements.");  
 return;  
 }  
   
 int smallest = int.MaxValue;  
 int secondSmallest = int.MaxValue;  
 int largest = int.MinValue;  
 int secondLargest = int.MinValue;  
   
 foreach (int num in numbers)  
 {  
 if (num < smallest)  
 {  
 secondSmallest = smallest;  
 smallest = num;  
 }  
 else if (num < secondSmallest && num != smallest)  
 {  
 secondSmallest = num;  
 }  
   
 if (num > largest)  
 {  
 secondLargest = largest;  
 largest = num;  
 }  
 else if (num > secondLargest && num != largest)  
 {  
 secondLargest = num;  
 }  
 }  
   
 if (secondSmallest == int.MaxValue)  
 {  
 Console.WriteLine("Second smallest does not exist.");  
 }  
 else  
 {  
 Console.WriteLine("Second smallest: " + secondSmallest);  
 }  
   
 if (secondLargest == int.MinValue)  
 {  
 Console.WriteLine("Second largest does not exist.");  
 }  
 else  
 {  
 Console.WriteLine("Second largest: " + secondLargest);  
 }  
 }  
}

### **FINAL Output**



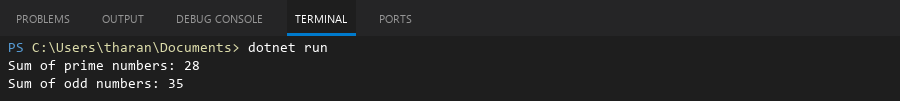
## **QUESTION 12**

***12. Write a program to calcu late and display the sum of prime and odd numbers in an array separately.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[] numbers = { 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 };  
 int primeSum = 0;  
 int oddSum = 0;  
  
 foreach (int num in numbers)  
 {  
 if (num % 2 != 0)  
 {  
 oddSum += num;  
 }  
  
 if (num > 1)  
 {  
 bool isPrime = true;  
 for (int i = 2; i \* i <= num; i++)  
 {  
 if (num % i == 0)  
 {  
 isPrime = false;  
 break;  
 }  
 }  
 if (isPrime)  
 {  
 primeSum += num;  
 }  
 }  
 }  
  
 Console.WriteLine("Sum of prime numbers: " + primeSum);  
 Console.WriteLine("Sum of odd numbers: " + oddSum);  
 }  
}

### **FINAL Output**



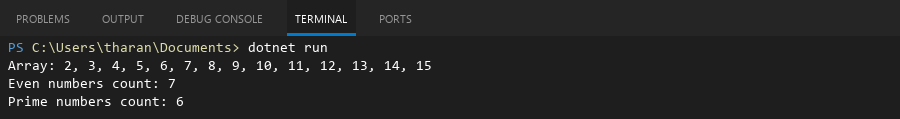
## **QUESTION 13**

***13. Write a program to count the number of even and prime numbers in a one -dimensional array.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[] numbers = { 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 };  
 int evenCount = 0;  
 int primeCount = 0;  
  
 foreach (int num in numbers)  
 {  
 if (num % 2 == 0)  
 {  
 evenCount++;  
 }  
  
 if (IsPrime(num))  
 {  
 primeCount++;  
 }  
 }  
  
 Console.WriteLine("Array: " + string.Join(", ", numbers));  
 Console.WriteLine("Even numbers count: " + evenCount);  
 Console.WriteLine("Prime numbers count: " + primeCount);  
 }  
  
 static bool IsPrime(int number)  
 {  
 if (number <= 1) return false;  
 if (number == 2) return true;  
 if (number % 2 == 0) return false;  
  
 for (int i = 3; i <= Math.Sqrt(number); i += 2)  
 {  
 if (number % i == 0)  
 return false;  
 }  
 return true;  
 }  
}

### **FINAL Output**



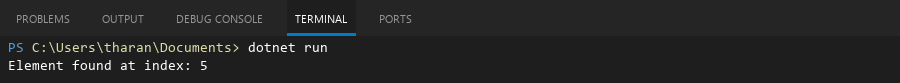
## **QUESTION 14**

***14. Implement a program to search for a specific element in an array using binary search .***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[] array = { 2, 5, 8, 12, 16, 23, 38, 45, 67, 89 };  
 int target = 23;  
 int result = BinarySearch(array, target);  
 if (result != -1)  
 {  
 Console.WriteLine("Element found at index: " + result);  
 }  
 else  
 {  
 Console.WriteLine("Element not found in the array.");  
 }  
 }  
  
 static int BinarySearch(int[] arr, int x)  
 {  
 int left = 0;  
 int right = arr.Length - 1;  
 while (left <= right)  
 {  
 int mid = left + (right - left) / 2;  
 if (arr[mid] == x)  
 {  
 return mid;  
 }  
 if (arr[mid] < x)  
 {  
 left = mid + 1;  
 }  
 else  
 {  
 right = mid - 1;  
 }  
 }  
 return -1;  
 }  
}

### **FINAL Output**



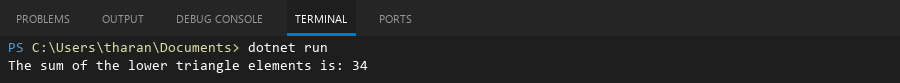
## **QUESTION 15**

***15. Write a program to calculate the sum of the lower triangle elements of a square matrix.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[,] matrix = {  
 {1, 2, 3},  
 {4, 5, 6},  
 {7, 8, 9}  
 };  
 int size = matrix.GetLength(0);  
 int sum = 0;  
   
 for (int i = 0; i < size; i++)  
 {  
 for (int j = 0; j <= i; j++)  
 {  
 sum += matrix[i, j];  
 }  
 }  
   
 Console.WriteLine("The sum of the lower triangle elements is: " + sum);  
 }  
}

### **FINAL Output**



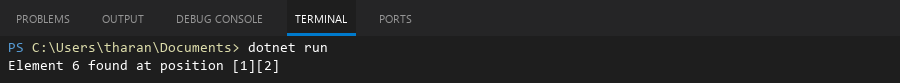
## **QUESTION 16**

***16. Write a C# program to perform linear search on a sorted jagged array .***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[][] jaggedArray = new int[][]  
 {  
 new int[] { 1, 3, 5 },  
 new int[] { 2, 4, 6, 8 },  
 new int[] { 10, 20 }  
 };  
  
 int target = 6;  
 bool found = false;  
 int row = -1;  
 int col = -1;  
  
 for (int i = 0; i < jaggedArray.Length; i++)  
 {  
 for (int j = 0; j < jaggedArray[i].Length; j++)  
 {  
 if (jaggedArray[i][j] == target)  
 {  
 found = true;  
 row = i;  
 col = j;  
 break;  
 }  
 }  
 if (found)  
 {  
 break;  
 }  
 }  
  
 if (found)  
 {  
 Console.WriteLine("Element " + target + " found at position [" + row + "][" + col + "]");  
 }  
 else  
 {  
 Console.WriteLine("Element " + target + " not found in the jagged array.");  
 }  
 }  
}

### **FINAL Output**



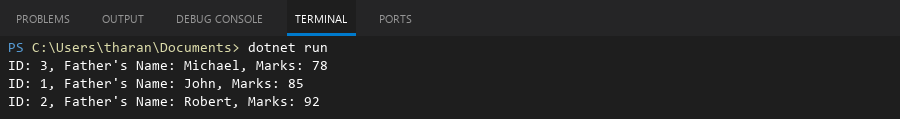
## **QUESTION 17**

***17. Create a Student class with properties (ID, Father\_ Name, Marks). Store multiple students in an object array and sort them by Marks. Further, s tore sorted students in a Linked List<T> and display them.***

### **Code Solution**

using System;  
using System.Collections.Generic;  
  
public class Student  
{  
 public int ID { get; set; }  
 public string Father\_Name { get; set; }  
 public int Marks { get; set; }  
}  
  
class Program  
{  
 static void Main()  
 {  
 Student[] studentsArray = new Student[]  
 {  
 new Student { ID = 1, Father\_Name = "John", Marks = 85 },  
 new Student { ID = 2, Father\_Name = "Robert", Marks = 92 },  
 new Student { ID = 3, Father\_Name = "Michael", Marks = 78 }  
 };  
  
 Array.Sort(studentsArray, (s1, s2) => s1.Marks.CompareTo(s2.Marks));  
  
 LinkedList<Student> sortedStudents = new LinkedList<Student>(studentsArray);  
  
 foreach (Student student in sortedStudents)  
 {  
 Console.WriteLine($"ID: {student.ID}, Father's Name: {student.Father\_Name}, Marks: {student.Marks}");  
 }  
 }  
}

### **FINAL Output**



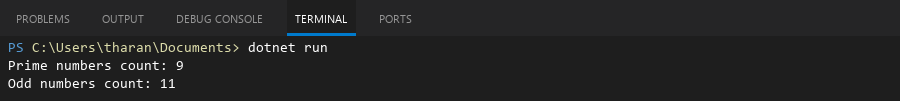
## **QUESTION 18**

***18. Write a program to count the number of prime and odd numbers in a one -dimensional array.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[] numbers = { 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 17, 19, 21, 23 };  
 int primeCount = 0;  
 int oddCount = 0;  
  
 foreach (int num in numbers)  
 {  
 if (num % 2 != 0)  
 {  
 oddCount++;  
 }  
  
 if (IsPrime(num))  
 {  
 primeCount++;  
 }  
 }  
  
 Console.WriteLine("Prime numbers count: " + primeCount);  
 Console.WriteLine("Odd numbers count: " + oddCount);  
 }  
  
 static bool IsPrime(int number)  
 {  
 if (number <= 1) return false;  
 if (number == 2) return true;  
 if (number % 2 == 0) return false;  
  
 for (int i = 3; i <= Math.Sqrt(number); i += 2)  
 {  
 if (number % i == 0)  
 return false;  
 }  
 return true;  
 }  
}

### **FINAL Output**



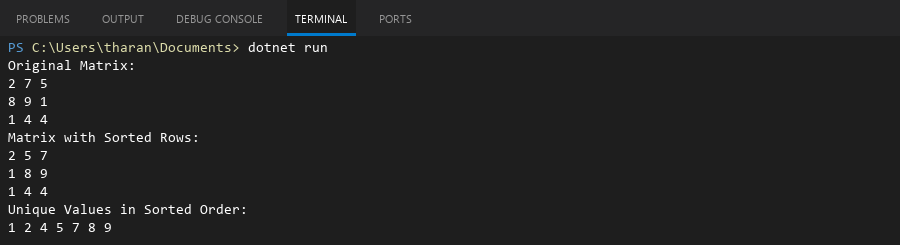
## **QUESTION 19**

***19. Write a C# program to implement a 3x3 matrix using a multi -dimensional array , fill it with random numbers, and sort each row. Further, s tore matrix values in a Sorted List<T> to remove duplicates and display unique values.***

### **Code Solution**

using System;  
using System.Collections.Generic;  
  
class Program  
{  
 static void Main()  
 {  
 int[,] matrix = new int[3, 3];  
 Random random = new Random();  
   
 for (int i = 0; i < 3; i++)  
 {  
 for (int j = 0; j < 3; j++)  
 {  
 matrix[i, j] = random.Next(1, 10);  
 }  
 }  
   
 Console.WriteLine("Original Matrix:");  
 for (int i = 0; i < 3; i++)  
 {  
 for (int j = 0; j < 3; j++)  
 {  
 Console.Write(matrix[i, j] + " ");  
 }  
 Console.WriteLine();  
 }  
   
 for (int i = 0; i < 3; i++)  
 {  
 int[] row = new int[3];  
 for (int j = 0; j < 3; j++)  
 {  
 row[j] = matrix[i, j];  
 }  
 Array.Sort(row);  
 for (int j = 0; j < 3; j++)  
 {  
 matrix[i, j] = row[j];  
 }  
 }  
   
 Console.WriteLine("Matrix with Sorted Rows:");  
 for (int i = 0; i < 3; i++)  
 {  
 for (int j = 0; j < 3; j++)  
 {  
 Console.Write(matrix[i, j] + " ");  
 }  
 Console.WriteLine();  
 }  
   
 SortedList<int, int> sortedList = new SortedList<int, int>();  
 for (int i = 0; i < 3; i++)  
 {  
 for (int j = 0; j < 3; j++)  
 {  
 if (!sortedList.ContainsKey(matrix[i, j]))  
 {  
 sortedList.Add(matrix[i, j], matrix[i, j]);  
 }  
 }  
 }  
   
 Console.WriteLine("Unique Values in Sorted Order:");  
 foreach (var value in sortedList.Keys)  
 {  
 Console.Write(value + " ");  
 }  
 Console.WriteLine();  
 }  
}

### **FINAL Output**



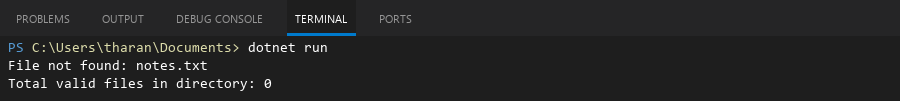
## **QUESTION 20**

***20. Write a C# program to implement a program that reads an array of filenames and searches for a specific file in the system. Further, s tore valid file names in a Directory collection and allow the user to retrieve details about a specific file.***

### **Code Solution**

using System;  
using System.Collections.Generic;  
using System.IO;  
  
class Program  
{  
 static void Main()  
 {  
 string[] filenames = { "data.txt", "report.pdf", "image.png", "notes.txt", "script.cs" };  
 string searchFile = "notes.txt";  
 Dictionary<string, FileInfo> validFiles = new Dictionary<string, FileInfo>();  
  
 foreach (string filename in filenames)  
 {  
 if (File.Exists(filename))  
 {  
 FileInfo fileInfo = new FileInfo(filename);  
 validFiles[filename] = fileInfo;  
 }  
 }  
  
 if (validFiles.ContainsKey(searchFile))  
 {  
 FileInfo file = validFiles[searchFile];  
 Console.WriteLine("File found: " + searchFile);  
 Console.WriteLine("Full path: " + file.FullName);  
 Console.WriteLine("Size: " + file.Length + " bytes");  
 Console.WriteLine("Last modified: " + file.LastWriteTime);  
 }  
 else  
 {  
 Console.WriteLine("File not found: " + searchFile);  
 }  
  
 Console.WriteLine("Total valid files in directory: " + validFiles.Count);  
 }  
}

### **FINAL Output**



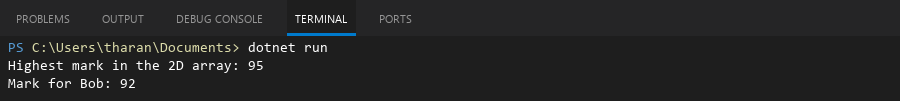
## **QUESTION 21**

***21. Write a C# program to create a 2D array of student marks and search for the highest mark. Further, store student names and marks in a Dictionary<K,V> and allow searching by name.***

### **Code Solution**

using System;  
using System.Collections.Generic;  
  
class Program  
{  
 static void Main()  
 {  
 int[,] marksArray = {  
 {85, 90, 78},  
 {92, 88, 95},  
 {70, 75, 80}  
 };  
  
 int highestMark = marksArray[0, 0];  
 for (int i = 0; i < marksArray.GetLength(0); i++)  
 {  
 for (int j = 0; j < marksArray.GetLength(1); j++)  
 {  
 if (marksArray[i, j] > highestMark)  
 {  
 highestMark = marksArray[i, j];  
 }  
 }  
 }  
  
 Console.WriteLine("Highest mark in the 2D array: " + highestMark);  
  
 Dictionary<string, int> studentMarks = new Dictionary<string, int>();  
 studentMarks.Add("Alice", 85);  
 studentMarks.Add("Bob", 92);  
 studentMarks.Add("Charlie", 70);  
  
 string searchName = "Bob";  
 if (studentMarks.ContainsKey(searchName))  
 {  
 Console.WriteLine("Mark for " + searchName + ": " + studentMarks[searchName]);  
 }  
 else  
 {  
 Console.WriteLine("Student " + searchName + " not found.");  
 }  
 }  
}

### **FINAL Output**



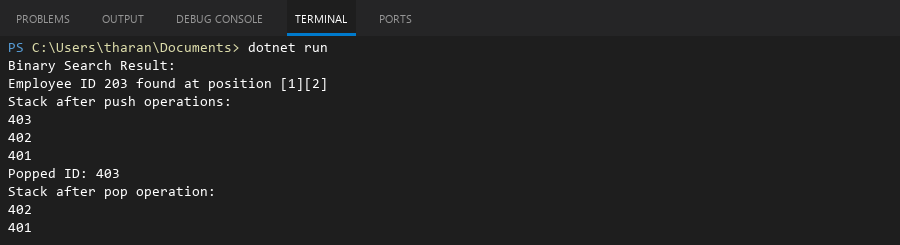
## **QUESTION 22**

***22. Write a C# program to implement Binary Search in a jagged array of employee IDs. Further, s tore IDs in a Stack<T> , push/pop operations for LIFO retrieval.***

### **Code Solution**

using System;  
using System.Collections.Generic;  
  
class Program  
{  
 static void Main()  
 {  
 int[][] jaggedArray = new int[][]  
 {  
 new int[] { 101, 102, 103 },  
 new int[] { 201, 202, 203, 204 },  
 new int[] { 301, 302 }  
 };  
  
 int searchId = 203;  
 bool found = false;  
 int rowIndex = -1;  
 int colIndex = -1;  
  
 for (int i = 0; i < jaggedArray.Length; i++)  
 {  
 int left = 0;  
 int right = jaggedArray[i].Length - 1;  
 while (left <= right)  
 {  
 int mid = left + (right - left) / 2;  
 if (jaggedArray[i][mid] == searchId)  
 {  
 found = true;  
 rowIndex = i;  
 colIndex = mid;  
 break;  
 }  
 if (jaggedArray[i][mid] < searchId)  
 {  
 left = mid + 1;  
 }  
 else  
 {  
 right = mid - 1;  
 }  
 }  
 if (found) break;  
 }  
  
 Console.WriteLine("Binary Search Result:");  
 if (found)  
 {  
 Console.WriteLine($"Employee ID {searchId} found at position [{rowIndex}][{colIndex}]");  
 }  
 else  
 {  
 Console.WriteLine($"Employee ID {searchId} not found.");  
 }  
  
 Stack<int> idStack = new Stack<int>();  
 idStack.Push(401);  
 idStack.Push(402);  
 idStack.Push(403);  
  
 Console.WriteLine("Stack after push operations:");  
 foreach (int id in idStack)  
 {  
 Console.WriteLine(id);  
 }  
  
 int poppedId = idStack.Pop();  
 Console.WriteLine($"Popped ID: {poppedId}");  
  
 Console.WriteLine("Stack after pop operation:");  
 foreach (int id in idStack)  
 {  
 Console.WriteLine(id);  
 }  
 }  
}

### **FINAL Output**



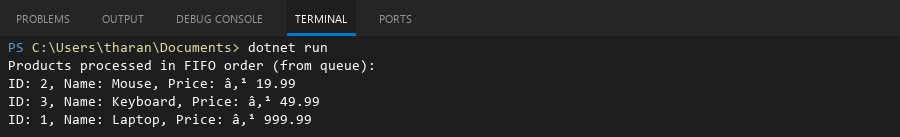
## **QUESTION 23**

***23. Write a C# program to create a Product class (ID, Name, Price) and store o bjects in an array by price. Further, u se a Queue<T> to manage product processing (FIFO order).***

### **Code Solution**

using System;  
using System.Collections.Generic;  
  
class Product  
{  
 public int ID { get; set; }  
 public string Name { get; set; }  
 public decimal Price { get; set; }  
}  
  
class Program  
{  
 static void Main()  
 {  
 Product[] products = new Product[]  
 {  
 new Product { ID = 1, Name = "Laptop", Price = 999.99m },  
 new Product { ID = 2, Name = "Mouse", Price = 19.99m },  
 new Product { ID = 3, Name = "Keyboard", Price = 49.99m }  
 };  
  
 Array.Sort(products, (x, y) => x.Price.CompareTo(y.Price));  
  
 Queue<Product> productQueue = new Queue<Product>();  
  
 foreach (Product product in products)  
 {  
 productQueue.Enqueue(product);  
 }  
  
 Console.WriteLine("Products processed in FIFO order (from queue):");  
 while (productQueue.Count > 0)  
 {  
 Product processedProduct = productQueue.Dequeue();  
 Console.WriteLine($"ID: {processedProduct.ID}, Name: {processedProduct.Name}, Price: {processedProduct.Price:C}");  
 }  
 }  
}

### **FINAL Output**



## **QUESTION 24**

***24. Write a program to calculate the sum of the diagonal elements of a square matrix.***

### **Code Solution**

using System;  
  
class Program  
{  
 static void Main()  
 {  
 int[,] matrix = {  
 {1, 2, 3},  
 {4, 5, 6},  
 {7, 8, 9}  
 };  
   
 int sum = 0;  
 int size = matrix.GetLength(0);  
   
 for (int i = 0; i < size; i++)  
 {  
 sum += matrix[i, i];  
 }  
   
 Console.WriteLine("Sum of diagonal elements: " + sum);  
 }  
}

### **FINAL Output**

