



International University of Business Agriculture and Technology

Department: Computer Science and Engineering

Semester: Spring 2025

Course Name: Visual Programming

Course Code: CSC 440

Section: A

Assignment topic: Lab task 01

Submitted To:

Suhala Lamia

Lecturer

Department of Computer Science and Engineering

Submitted By:

Samiul Karim Mazumder

22303308

Date of Submission: 2/03/25

Experiment 01: Write a program that will print all the perfect numbers from a user-defined range.

Objective: The objective of this program is to find and display all perfect numbers within a given range defined by the user. A perfect number is a positive integer that is equal to the sum of its proper divisors, excluding itself.

Algorithm:

1. **Start**
2. Ask the user to **enter the start and end of the range**.
3. Loop through all numbers in the range:
 - Check if each number is a **perfect number** using the `IsPerfect()` function.
4. Inside `IsPerfect()`:
 - Initialize `sum = 1` (since `1` is a divisor for all numbers).
 - Iterate from `2` to `num/2` to find all divisors.
 - If the sum of the divisors equals the number itself, it is a **perfect number**.
5. Print all perfect numbers found.
6. **End**

Program:

```
using System;
```

```
namespace lab1
```

```
{
```

```
    class Program
```

```
    {
```

```
        static void Main(string[] args)
```

```
        {
```

```
            Console.WriteLine("Enter the start of the range:");
```

```
            int start = int.Parse(Console.ReadLine());
```

```
Console.WriteLine("Enter the end of the range:");  
int end = int.Parse(Console.ReadLine());
```

```
Console.WriteLine($"Perfect numbers between {start} and {end}:");  
FindPerfectNumbers(start, end);  
}
```

```
static void FindPerfectNumbers(int start, int end)  
{  
    for (int num = start; num <= end; num++)  
    {  
        if (IsPerfect(num))  
        {  
            Console.WriteLine(num);  
        }  
    }  
}
```

```
static bool IsPerfect(int num)  
{  
    if (num < 2) return false;  
  
    int sum = 1;  
    for (int i = 2; i <= num / 2; i++)  
    {  
        if (num % i == 0)  
        {  
            sum += i;  
        }  
    }  
    return sum == num;  
}  
}
```

Output:

A screenshot of the Microsoft Visual Studio Debug Console window. The window has a title bar with the Visual Studio logo and the text "Microsoft Visual Studio Debug Console". The console area is light gray and contains the following text: "Enter the start of the range:", "1", "Enter the end of the range:", "1000", "Perfect numbers between 1 and 1000:", "6", "28", "496", "C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1\bin\Debug\net8.0\lab1.1.exe (process 15236) exited with code 0 (0x0).", "To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.", and "Press any key to close this window . . .".

```
Microsoft Visual Studio Debug Console
Enter the start of the range:
1
Enter the end of the range:
1000
Perfect numbers between 1 and 1000:
6
28
496
C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1\bin\Debug\net8.0\lab1.1.exe (process 15236)
exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console
when debugging stops.
Press any key to close this window . . .
```

Experiment 02: Write a program to implement the ATM transaction system

Features:

a) The user can withdraw money (less than the total amount)

per transaction user gets the amount which is divisible by 500. and the minimum amount in the account without withdrawal is 500

b) The user can check the balance

Objective: The objective of this program is to simulate an ATM Transaction System where users can:

1. Withdraw money (amount must be a multiple of 500, and a minimum balance of 500 must be maintained).
2. Check account balance to see the current available amount.

This ensures proper banking constraints while allowing users to perform secure transactions.

Algorithm:

1. **Start**
2. Initialize the **account balance** (e.g., 10,000).
3. Display an **ATM menu** with the following options:
 - **Withdraw Money**
 - **Check Balance**
 - **Exit**
4. If the user selects **Withdraw Money**:
 - Ask the user to enter the withdrawal amount.
 - Validate that:
 - The amount is a **multiple of 500**.
 - The **remaining balance is at least 500** after withdrawal.
 - If valid, deduct the amount and display the new balance.
 - If invalid, show an error message.
5. If the user selects **Check Balance**, display the current account balance.
6. If the user selects **Exit**, End the program.
7. Repeat until the user chooses to exit.
8. **End**

Program:

using System;

namespace ATMSystem

{

class Program

{

static void Main(string[] args)

{

ATM atm = new ATM(10000);

```
        atm.Menu();  
    }  
}
```

```
class ATM
```

```
{  
    private int balance;  
  
    public ATM(int initialBalance)  
    {  
        balance = initialBalance;  
    }  
}
```

```
public void Menu()  
{  
    while (true)  
    {  
        Console.WriteLine("\n===== ATM Menu =====");  
        Console.WriteLine("1. Withdraw Money");  
        Console.WriteLine("2. Check Balance");  
        Console.WriteLine("3. Exit");  
        Console.Write("Select an option: ");  
    }  
}
```

```
string choice = Console.ReadLine();  
switch (choice)  
{  
    case "1":  
        WithdrawMoney();  
        break;  
    case "2":  
        CheckBalance();  
        break;  
    case "3":  
        Console.WriteLine("Thank you for using the ATM!");  
        return;  
    default:  
        Console.WriteLine("Invalid option.");  
        break;  
}  
}
```

```
private void WithdrawMoney()  
{  
    Console.Write("\nEnter withdrawal amount: ");  
    int amount;
```

```
if (!int.TryParse(Console.ReadLine(), out amount) || amount <= 0)
{
    Console.WriteLine("Invalid amount. Please enter a valid number.");
    return;
}

if (amount % 500 != 0)
{
    Console.WriteLine("Error: Amount must be a multiple of 500.");
    return;
}

if (balance - amount < 500)
{
    Console.WriteLine("Error: Insufficient balance. Minimum balance of
500 must be maintained.");
    return;
}

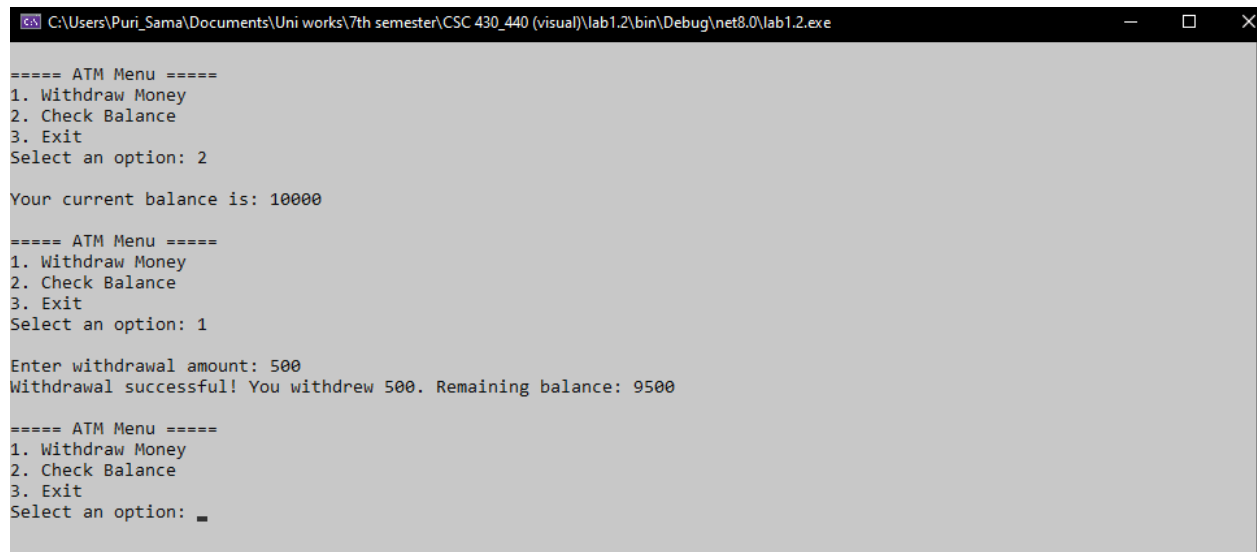
balance -= amount;

Console.WriteLine($"Withdrawal successful! You withdrew {amount}.
Remaining balance: {balance}");
}
```



```
private void CheckBalance()
{
    Console.WriteLine($"
Your current balance is: {balance}");
}
}
```

Output:



```
C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.2\bin\Debug\net8.0\lab1.2.exe

===== ATM Menu =====
1. Withdraw Money
2. Check Balance
3. Exit
Select an option: 2

Your current balance is: 10000

===== ATM Menu =====
1. Withdraw Money
2. Check Balance
3. Exit
Select an option: 1

Enter withdrawal amount: 500
Withdrawal successful! You withdrew 500. Remaining balance: 9500

===== ATM Menu =====
1. Withdraw Money
2. Check Balance
3. Exit
Select an option: _
```

Experiment 3: Write a program and identify which Chinese zodiac year your birthday is.

Objective:

The objective of this program is to determine the Chinese Zodiac sign based on the user's birth year. The Chinese Zodiac cycle consists of 12 animals, and each year is associated with a specific animal. The program will take the user's birth year as input and determine their corresponding Chinese Zodiac sign using modulus operation.

Algorithm:

1. **Start**
2. Prompt the user to **enter their birth year**.
3. Validate the input to ensure it is a **positive integer**.
4. Store the **Chinese Zodiac animal names** in an **array** in the correct order.
5. Calculate the **Zodiac sign index** using:
 - $\text{index} = \text{birthYear} \% 12$ [% = Modulus]
6. Retrieve the corresponding **Zodiac animal** from the array.
7. Display the **Chinese Zodiac sign** to the user.
8. **End**

Program:

```
using System;
```

```
namespace ChineseZodiac
{
    class Program
    {
        static void Main(string[] args)
        {
```

```

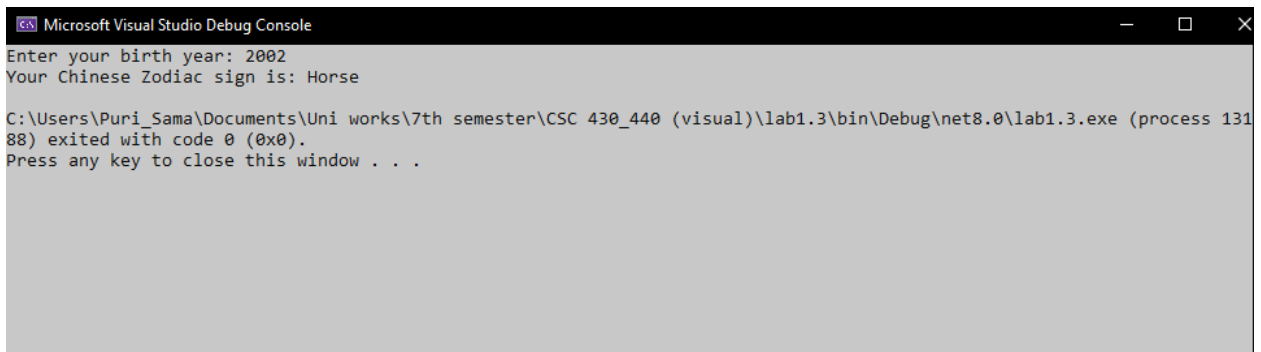
Console.Write("Enter your birth year: ");
int birthYear;
if (!int.TryParse(Console.ReadLine(), out birthYear) || birthYear <= 0)
{
    Console.WriteLine("Invalid input! Please enter a valid positive year.");
    return;
}

string[] zodiacAnimals = {
    "Monkey", "Rooster", "Dog", "Pig", "Rat", "Ox",
    "Tiger", "Rabbit", "Dragon", "Snake", "Horse", "Sheep"
};
int index = birthYear % 12;
string zodiacSign = zodiacAnimals[index];

Console.WriteLine($"Your Chinese Zodiac sign is: {zodiacSign}");
}
}
}

```

Output:



```

Microsoft Visual Studio Debug Console
Enter your birth year: 2002
Your Chinese Zodiac sign is: Horse

C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.3\bin\Debug\net8.0\lab1.3.exe (process 13188) exited with code 0 (0x0).
Press any key to close this window . . .

```

Experiment 4: Write a program to sort a given array in the ascending order.

Objective

The objective of this program is to take a **user-defined array**, sort it in **ascending order**, and display the sorted array. The program uses the **Array.Sort() method** in C# to achieve this efficiently.

Algorithm

1. **Start**
2. Prompt the user to **enter the number of elements** in the array.
3. Create an **array of the given size**.
4. Ask the user to **input each element** of the array.
5. Use the **Array.Sort() method** to sort the array in **ascending order**.
6. Display the **sorted array**.
7. **End**

Program:

```
using System;

namespace ArraySorting
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.Write("Enter the number of elements: ");
            int size = int.Parse(Console.ReadLine());

            int[] arr = new int[size];

            Console.WriteLine("Enter the elements:");
```

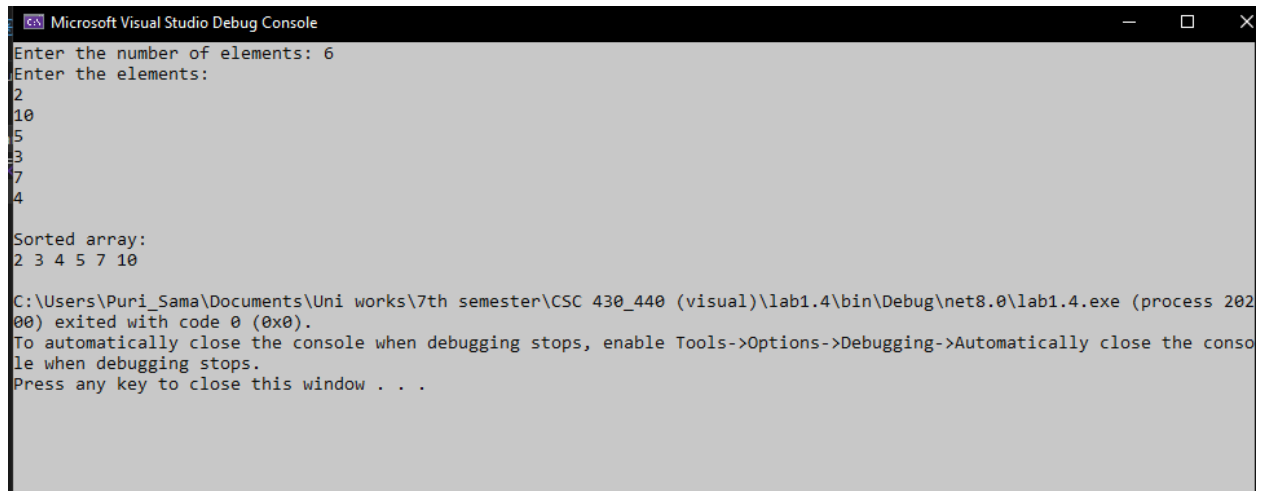
```
        for (int i = 0; i < size; i++)
            arr[i] = int.Parse(Console.ReadLine());

        Array.Sort(arr);

        Console.WriteLine("\nSorted array:");
        foreach (int num in arr)
            Console.Write(num + " ");

        Console.WriteLine();
    }
}
```

Output:



The screenshot shows the Microsoft Visual Studio Debug Console window. The output of the program is as follows:

```
Enter the number of elements: 6
Enter the elements:
2
10
5
3
7
4

Sorted array:
2 3 4 5 7 10

C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.4\bin\Debug\net8.0\lab1.4.exe (process 20200) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Experiment 5: Write a program where input type of the shape output is the area of that shape.

Objective

The objective of this program is to calculate the **area of different shapes** (circle, square, rectangle) based on the user's input for the type of shape and its dimensions. The program will ask for the **shape type** and then compute and output the **area**.

Algorithm

1. **Start**
2. Ask the user to input the **shape type** (circle, square, or rectangle).
3. Based on the input, ask for the required dimensions:
 - For **circle**, ask for the **radius**.
 - For **square**, ask for the **side length**.
 - For **rectangle**, ask for **length and width**.
4. Compute the **area** for the chosen shape:
 - Circle: $\text{Area} = \pi * \text{radius}^2$
 - Square: $\text{Area} = \text{side} * \text{side}$
 - Rectangle: $\text{Area} = \text{length} * \text{width}$
5. Output the **calculated area**.
6. **End**

Program:

using System;

namespace ShapeArea

{

class Program

{

```
static void Main(string[] args)
{
    Console.Write("Enter the shape type (circle, square, rectangle): ");
    string shapeType = Console.ReadLine().ToLower();

    if (shapeType == "circle")
    {
        Console.Write("Enter the radius: ");
        double radius = double.Parse(Console.ReadLine());
        double area = Math.PI * radius * radius;
        Console.WriteLine("Area of the circle: " + area);
    }
    else if (shapeType == "square")
    {
        Console.Write("Enter the side length: ");
        double side = double.Parse(Console.ReadLine());
        double area = side * side;
        Console.WriteLine("Area of the square: " + area);
    }
    else if (shapeType == "rectangle")
    {
        Console.Write("Enter the length: ");
        double length = double.Parse(Console.ReadLine());
```

```

        Console.WriteLine("Enter the width: ");

        double width = double.Parse(Console.ReadLine());

        double area = length * width;

        Console.WriteLine("Area of the rectangle: " + area);

    }

    else

    {

        Console.WriteLine("Invalid shape type!");

    }

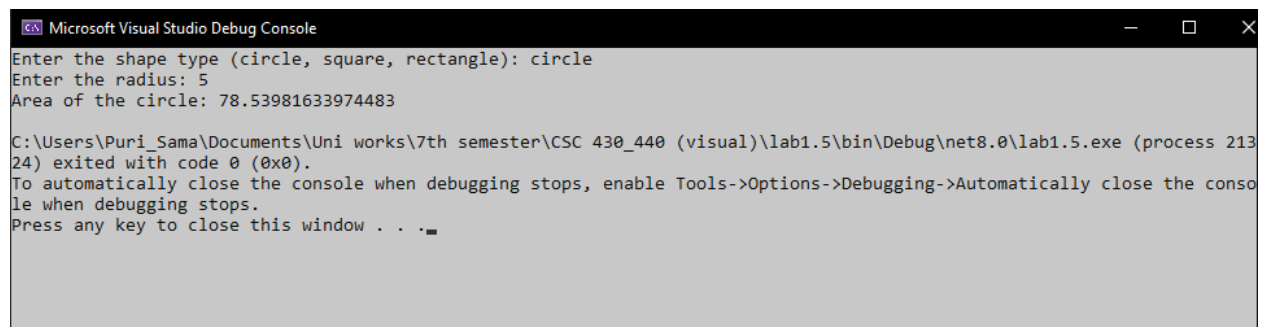
}

}

}

```

Output:

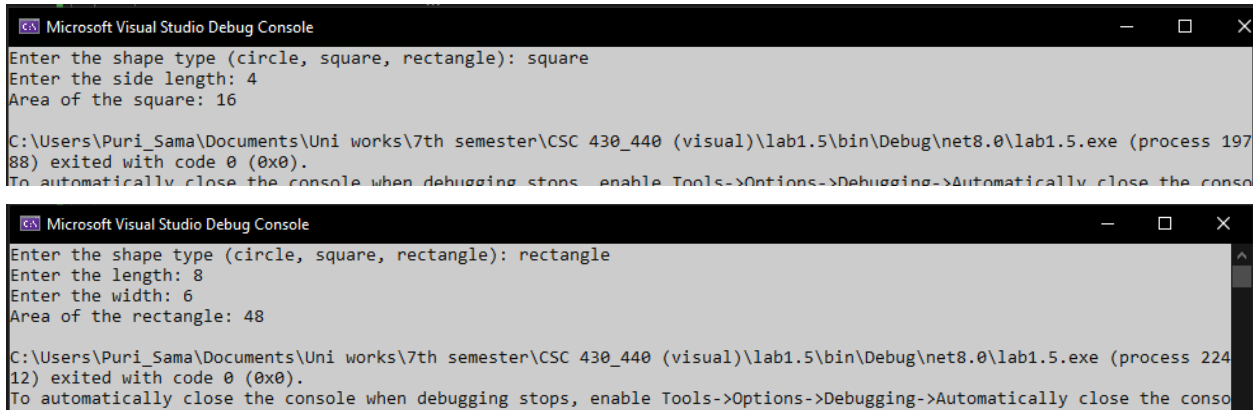


```

Microsoft Visual Studio Debug Console
Enter the shape type (circle, square, rectangle): circle
Enter the radius: 5
Area of the circle: 78.53981633974483

C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.5\bin\Debug\net8.0\lab1.5.exe (process 21324) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .

```

The image contains two screenshots of the Microsoft Visual Studio Debug Console. The top screenshot shows the program running for a square: the user enters 'square' for the shape type and '4' for the side length, resulting in an area of 16. The bottom screenshot shows the program running for a rectangle: the user enters 'rectangle' for the shape type, '8' for the length, and '6' for the width, resulting in an area of 48. Both screenshots show the program path as C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.5\bin\Debug\net8.0\lab1.5.exe and indicate the process exited with code 0.

```
Microsoft Visual Studio Debug Console
Enter the shape type (circle, square, rectangle): square
Enter the side length: 4
Area of the square: 16

C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.5\bin\Debug\net8.0\lab1.5.exe (process 19788) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console window.

Microsoft Visual Studio Debug Console
Enter the shape type (circle, square, rectangle): rectangle
Enter the length: 8
Enter the width: 6
Area of the rectangle: 48

C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.5\bin\Debug\net8.0\lab1.5.exe (process 22412) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console window.
```

Experiment 6: Write a program to search an element from the given array

Objective: The objective of this program is to search for a specific element in a user-defined array. The program will take the array elements and the search element as input, and will return whether the element exists in the array.

Algorithm

1. Start
2. Ask the user to input the size of the array.
3. Take input for each element of the array.
4. Ask the user for the element to search.
5. Use a loop to check if the search element exists in the array.
6. If the element is found, print "Element found".
7. If the element is not found, print "Element not found".
8. End

Program:

```
using System;

namespace ArraySearch
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter the number of elements in the array: ");
            int size = int.Parse(Console.ReadLine());

            int[] arr = new int[size];

            Console.WriteLine("Enter the elements of the array:");
            for (int i = 0; i < size; i++)
                arr[i] = int.Parse(Console.ReadLine());

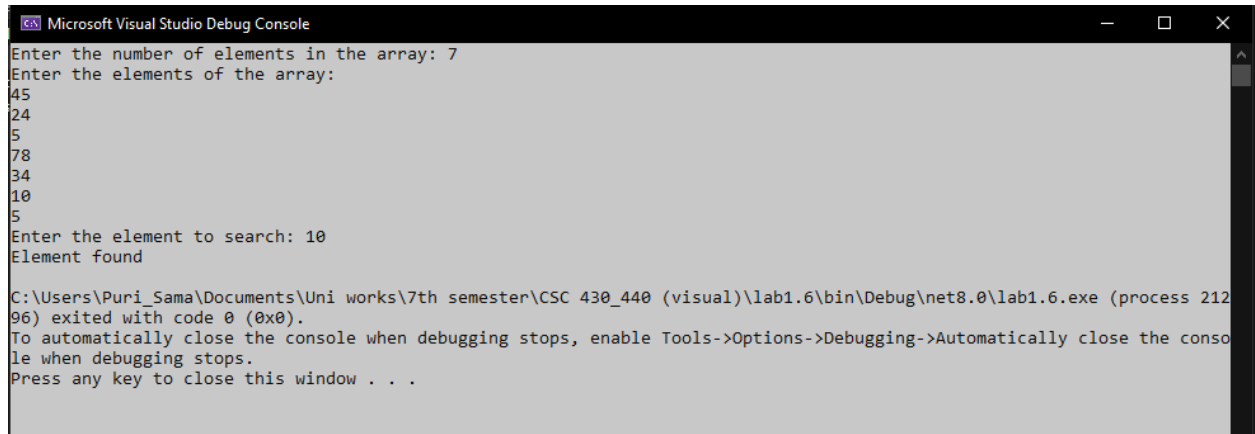
            Console.WriteLine("Enter the element to search: ");
            int searchElement = int.Parse(Console.ReadLine());

            bool found = false;
            foreach (int num in arr)
            {
                if (num == searchElement)
                {
                    found = true;
                    break;
                }
            }

            if (found)
                Console.WriteLine("Element found");
            else
```

```
        Console.WriteLine("Element not found");  
    }  
}  
}
```

Output:



The screenshot shows the Microsoft Visual Studio Debug Console window. The title bar reads "Microsoft Visual Studio Debug Console". The console output is as follows:

```
Enter the number of elements in the array: 7  
Enter the elements of the array:  
45  
24  
5  
78  
34  
10  
5  
Enter the element to search: 10  
Element found  
  
C:\Users\Puri_Sama\Documents\Uni works\7th semester\CSC 430_440 (visual)\lab1.6\bin\Debug\net8.0\lab1.6.exe (process 21296) exited with code 0 (0x0).  
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.  
Press any key to close this window . . .
```