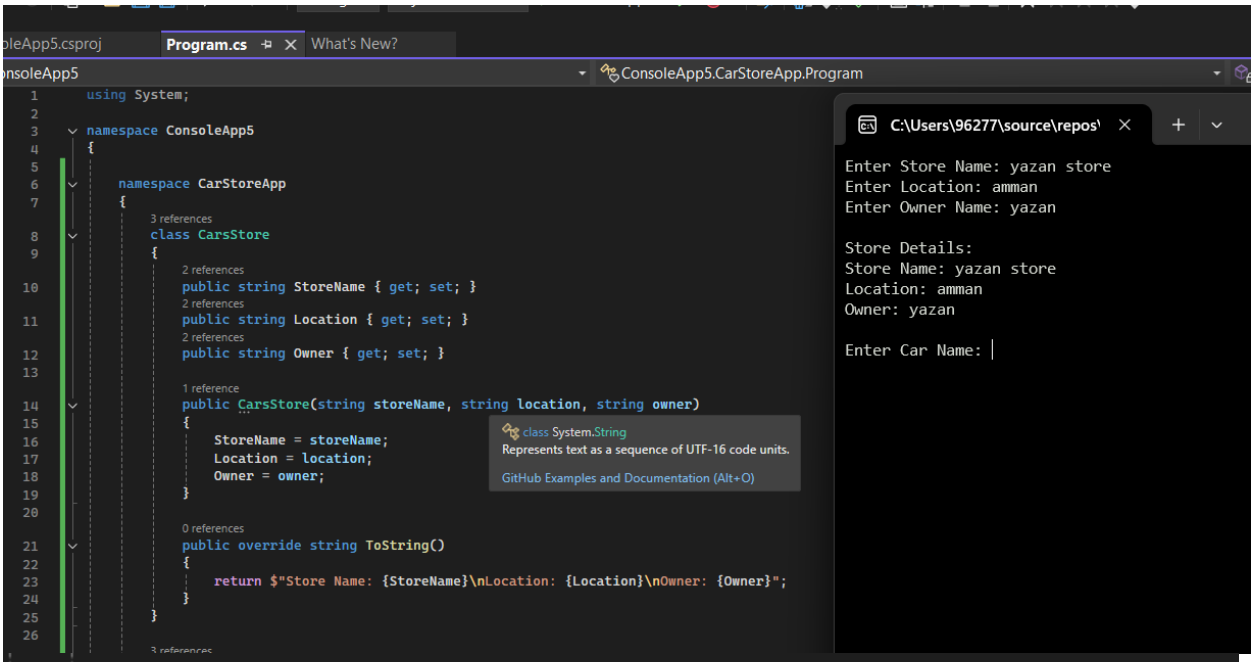
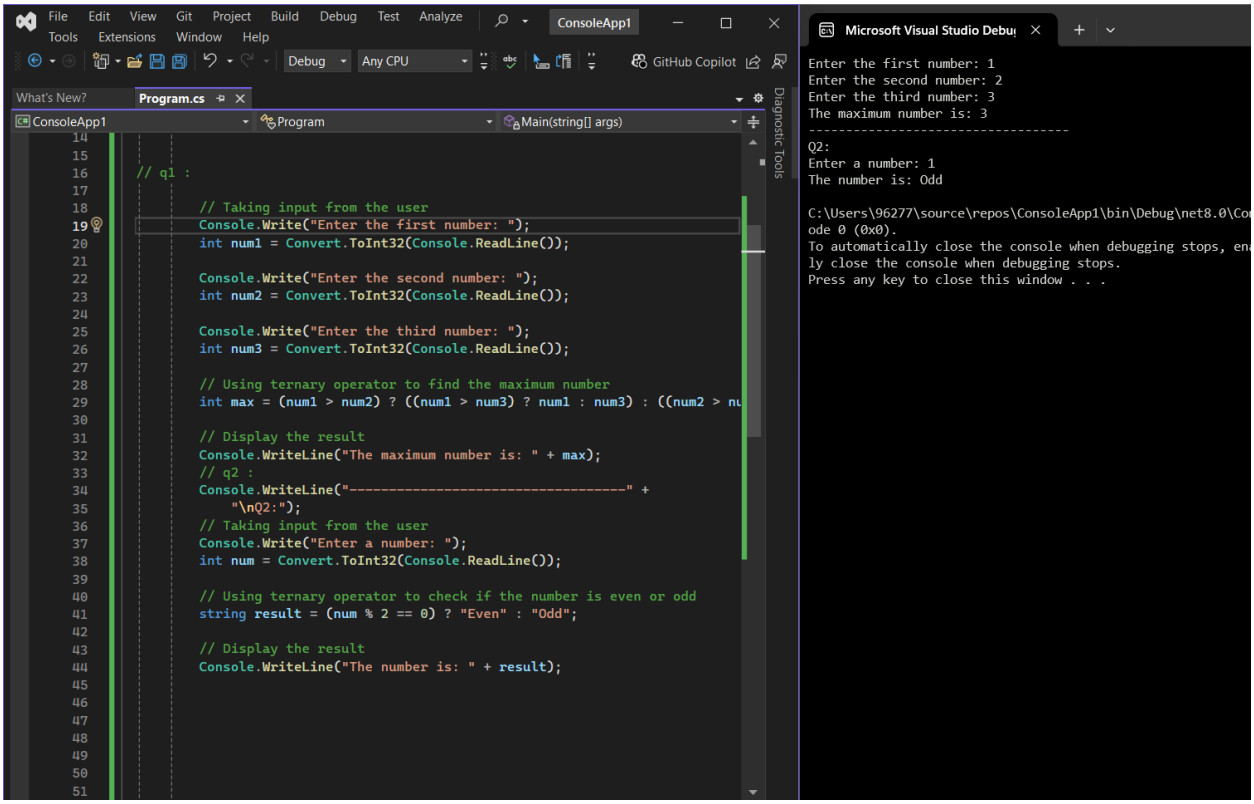
Ex 1.



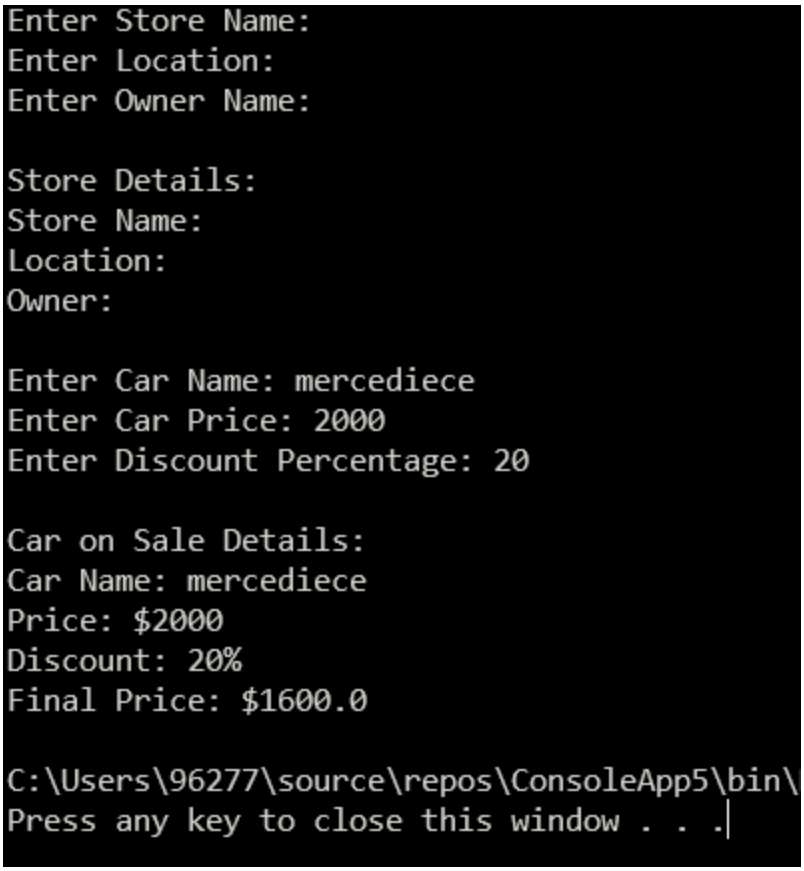

```
3 references
class Car
{
    2 references
    public string CarName { get; set; }
    4 references
    public decimal Price { get; set; }

    1 reference
    public Car(string carName, decimal price)
    {
        CarName = carName;
        Price = price;
    }

    2 references
    public override string ToString()
    {
        return $"Car Name: {CarName}\nPrice: ${Price}";
    }
}

3 references
```

```
3 references
class CarOnSale : Car
{
    3 references
    public double DiscountPercent { get; set; }
    2 references
    public decimal FinalPrice { get; private set; }

    1 reference
    public CarOnSale(string carName, decimal price, double discountPercent)
        : base(carName, price)
    {
        DiscountPercent = discountPercent;
        CalculateFinalPrice();
    }

    1 reference
    private void CalculateFinalPrice()
    {
        FinalPrice = Price - (Price * (decimal)(DiscountPercent / 100));
    }

    2 references
    public override string ToString()
    {
        return $"{base.ToString()}\nDiscount: {DiscountPercent}%\nFinal Price: ${FinalPrice}";
    }
}
```

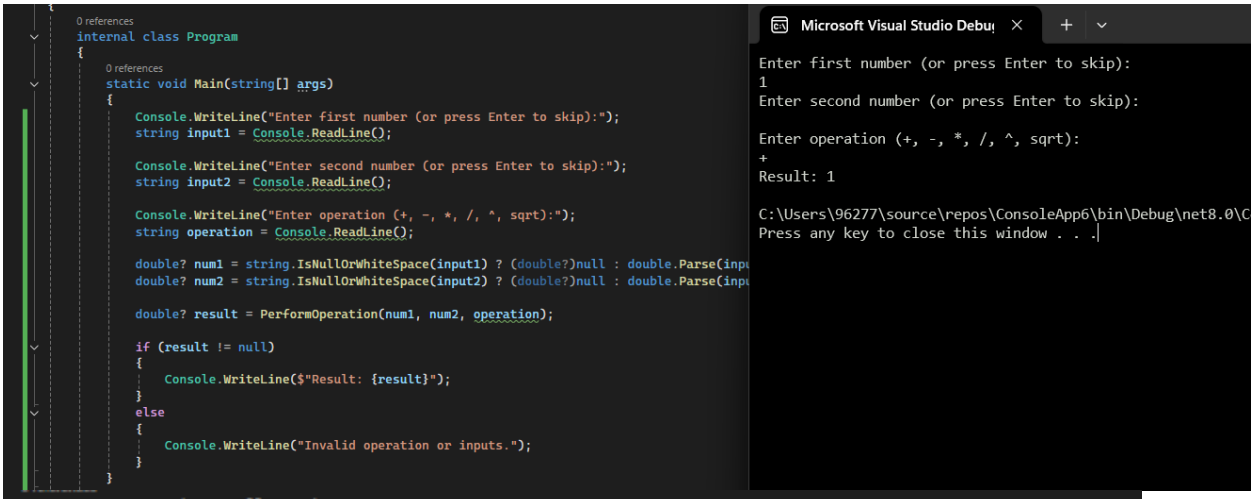
```
class Program
{
    0 references
    static void Main(string[] args)
    {
        // Get CarsStore details
        Console.Write("Enter Store Name: ");
        string storeName = Console.ReadLine();

        Console.Write("Enter Location: ");
        string location = Console.ReadLine();

        Console.Write("Enter Owner Name: ");
        string owner = Console.ReadLine();

        CarsStore store = new CarsStore(storeName, location, owner);
        Console.WriteLine("\nStore Details:");
        Console.WriteLine(store);

        // Get Car details
        Console.Write("\nEnter Car Name: ");
        string carName = Console.ReadLine();

        Console.Write("Enter Car Price: ");
        decimal carPrice = decimal.Parse(Console.ReadLine());

        Console.Write("Enter Discount Percentage: ");
        double discount = double.Parse(Console.ReadLine());

        CarOnSale carOnSale = new CarOnSale(carName, carPrice, discount);

        // Print car details
        Console.WriteLine("\nCar on Sale Details:");
        Console.WriteLine(carOnSale);
    }
}
```


File Edit View Git Project Build Debug Test Analyze
Tools Extensions Window Help
ConsoleApp1
Debug
Any CPU
GitHub Copilot
What's New?
Program.cs
ConsoleApp1
Program
Main(string[] args)
Diagnostic Tools
// q1 :
// Taking input from the user
Console.Write("Enter the first number: ");
int num1 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the second number: ");
int num2 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the third number: ");
int num3 = Convert.ToInt32(Console.ReadLine());
// Using ternary operator to find the maximum number
int max = (num1 > num2) ? ((num1 > num3) ? num1 : num3) : ((num2 > nu
// Display the result
Console.WriteLine("The maximum number is: " + max);
// q2 :
Console.WriteLine("----------------------------------" +
"\nQ2:");
// Taking input from the user
Console.Write("Enter a number: ");
int num = Convert.ToInt32(Console.ReadLine());
// Using ternary operator to check if the number is even or odd
string result = (num % 2 == 0) ? "Even" : "Odd";
// Display the result
Console.WriteLine("The number is: " + result);
Microsoft Visual Studio Debu
Enter the first number: 1
Enter the second number: 2
Enter the third number: 3
The maximum number is: 3

Q2:
Enter a number: 1
The number is: Odd
C:\Users\96277\source\repos\ConsoleApp1\bin\Debug\net8.0\Co
ode 0 (0x0).
To automatically close the console when debugging stops, en
ly close the console when debugging stops.
Press any key to close this window . . .


Enter Store Name:
Enter Location:
Enter Owner Name:
Store Details:
Store Name:
Location:
Owner:
Enter Car Name: mercediece
Enter Car Price: 2000
Enter Discount Percentage: 20
Car on Sale Details:
Car Name: mercediece
Price: $2000
Discount: 20%
Final Price: $1600.0
C:\Users\96277\source\repos\ConsoleApp5\bin\
Press any key to close this window . . .

Ex.2



```
0 references
internal class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("Enter first number (or press Enter to skip):");
        string input1 = Console.ReadLine();

        Console.WriteLine("Enter second number (or press Enter to skip):");
        string input2 = Console.ReadLine();

        Console.WriteLine("Enter operation (+, -, *, /, ^, sqrt):");
        string operation = Console.ReadLine();

        double? num1 = string.IsNullOrWhiteSpace(input1) ? (double?)null : double.Parse(inpu
        double? num2 = string.IsNullOrWhiteSpace(input2) ? (double?)null : double.Parse(inpu

        double? result = PerformOperation(num1, num2, operation);

        if (result != null)
        {
            Console.WriteLine($"Result: {result}");
        }
        else
        {
            Console.WriteLine("Invalid operation or inputs.");
        }
    }
```

```
Microsoft Visual Studio Debu
Enter first number (or press Enter to skip):
1
Enter second number (or press Enter to skip):

Enter operation (+, -, *, /, ^, sqrt):
+
Result: 1

C:\Users\96277\source\repos\ConsoleApp6\bin\Debug\net8.0\C
Press any key to close this window . . .
```

```
static void Main(string[] args)
{
    Console.WriteLine("Enter first number (or press Enter to skip):");
    string input1 = Console.ReadLine();

    Console.WriteLine("Enter second number (or press Enter to skip):");
    string input2 = Console.ReadLine();

    Console.WriteLine("Enter operation (+, -, *, /, ^, sqrt):");
    string operation = Console.ReadLine();

    double? num1 = string.IsNullOrWhiteSpace(input1) ? (double?)null : double.Parse(input1);
    double? num2 = string.IsNullOrWhiteSpace(input2) ? (double?)null : double.Parse(input2);

    double? result = PerformOperation(num1, num2, operation);

    if (result != null)
    {
        Console.WriteLine($"Result: {result}");
    }
    else
    {
        Console.WriteLine("Invalid operation or inputs.");
    }
}

1 reference
static double? PerformOperation(double? num1, double? num2, string operation)
{
    try
    {
        // Ensure values default to 0 if null
        double value1 = num1 ?? 0;
        double value2 = num2 ?? 0;

        return operation switch
        {
            "+" => value1 + value2,
            "-" => value1 - value2,
            "*" => value1 * value2,
            "/" => value2 != 0 ? value1 / value2 : double.NaN, // Prevent division by zero
            "^" => Math.Pow(value1, value2),
            "sqrt" => Math.Sqrt(value1), // Only considers the first number
            _ => null // Invalid operation
        };
    }
    catch
    {
        return null; // Return null if any exception occurs
    }
}
```