



REVA
UNIVERSITY
Bengaluru, India



C# Assignment

Shivam Jain

1. Write a program in C# to print a number if it is prime; otherwise display the largest factor of that number.

```
using System;
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Enter a number:");
        int number = Convert.ToInt32(Console.ReadLine());
        if (IsPrime(number))
        {
            Console.WriteLine($"{number} is a prime number.");
        }
        else
        {
            int largestFactor = GetLargestFactor(number);
            Console.WriteLine($"The largest factor of {number} is {largestFactor}.");
        }
    }
    static bool IsPrime(int n)
    {
        if (n <= 1)
            return false;
        for (int i = 2; i <= Math.Sqrt(n); i++)
        {
            if (n % i == 0)
                return false;
        }
        return true;
    }
    static int GetLargestFactor(int n)
    {
        int largestFactor = 1;
        for (int i = 2; i <= n / 2; i++)
        {
            if (n % i == 0)
                largestFactor = i;
        }
        return largestFactor;
    }
}
```

OUTPUT :

```
● shivam@fedora:~/Code/C#/C#assignment$ dotnet run
Enter a number:
32
The largest factor of 32 is 16.
```

2. Write a C# program for addition and multiplication of two matrices.

```
using System;
class MatrixOperations
{
    static void Main()
    {
        int[,] matrix1 = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };
        int[,] matrix2 = { { 9, 8, 7 }, { 6, 5, 4 }, { 3, 2, 1 } };
        Console.WriteLine("Matrix 1:");
        PrintMatrix(matrix1);
        Console.WriteLine("\nMatrix 2:");
        PrintMatrix(matrix2);
        Console.WriteLine("\nAddition of Matrix 1 and Matrix 2:");
        int[,] additionResult = AddMatrices(matrix1, matrix2);
        PrintMatrix(additionResult);
        Console.WriteLine("\nMultiplication of Matrix 1 and Matrix 2:");
        int[,] multiplicationResult = MultiplyMatrices(matrix1, matrix2);
        PrintMatrix(multiplicationResult);
    }
    static void PrintMatrix(int[,] matrix)
    {
        int rows = matrix.GetLength(0);
        int cols = matrix.GetLength(1);
        for (int i = 0; i < rows; i++)
        {
            for (int j = 0; j < cols; j++)
            {
                Console.Write(matrix[i, j] + "t");
            }
            Console.WriteLine();
        }
    }
    static int[,] AddMatrices(int[,] matrix1, int[,] matrix2)
    {
        int rows = matrix1.GetLength(0);
        int cols = matrix1.GetLength(1);
        int[,] result = new int[rows, cols];
        for (int i = 0; i < rows; i++)
        {
            for (int j = 0; j < cols; j++)
            {
                result[i, j] = matrix1[i, j] + matrix2[i, j];
            }
        }
        return result;
    }
}
```

```

}
static int[,] MultiplyMatrices(int[,] matrix1, int[,] matrix2)
{
    int rows1 = matrix1.GetLength(0);
    int cols1 = matrix1.GetLength(1);
    int cols2 = matrix2.GetLength(1);
    int[,] result = new int[rows1, cols2];
    for (int i = 0; i < rows1; i++)
    {
        for (int j = 0; j < cols2; j++)
        {
            int sum = 0;
            for (int k = 0; k < cols1; k++)
            {
                sum += matrix1[i, k] * matrix2[k, j];
            }
            result[i, j] = sum;
        }
    }
    return result;
}
}

```

OUTPUT

```

● shivam@fedora:~/Code/C#/C#assignment$ dotnet run
Matrix 1:
1      2      3
4      5      6
7      8      9

Matrix 2:
9      8      7
6      5      4
3      2      1

Addition of Matrix 1 and Matrix 2:
10     10     10
10     10     10
10     10     10

Multiplication of Matrix 1 and Matrix 2:
30     24     18
84     69     54
138    114    90

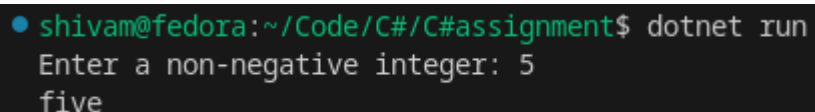
```

3. Write a C# program to display the digits of an integer in words.

```
using System;
public class DigitToWords
{
    public static void Main(string[] args)
    {
        // Array to store words for digits 0-9
        string[] digits = { "zero", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine"
    };

    // Get input integer
    Console.WriteLine("Enter a non-negative integer: ");
    int number = Convert.ToInt32(Console.ReadLine());
    // Check for negative number
    if (number < 0)
    {
        Console.WriteLine("Invalid input. Please enter a non-negative integer.");
        return;
    }
    // Handle zero case
    if (number == 0)
    {
        Console.WriteLine("The number is zero.");
        return;
    }
    // Process digits one by one
    while (number > 0)
    {
        int digit = number % 10;
        Console.Write(digits[digit] + " ");
        number /= 10;
    }
    Console.WriteLine();
}
}
```

OUTPUT

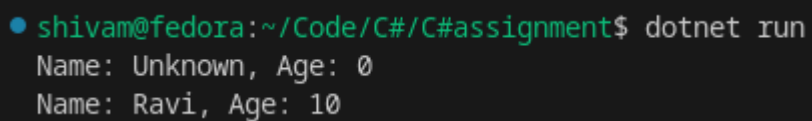


```
shivam@fedora:~/Code/C#/C#assignment$ dotnet run
Enter a non-negative integer: 5
five
```

4. Write a C# programs to demonstrate the concepts of Constructors and Inheritance

```
using System;
class Person
{
    public string Name { get; set; }
    public int Age { get; set; }
    // Default constructor (parameterless)
    public Person()
    {
        Name = "Unknown";
        Age = 0;
    }
    // Parameterized constructor
    public Person(string name, int age)
    {
        Name = name;
        Age = age;
    }
    public void DisplayInfo()
    {
        Console.WriteLine($"Name: {Name}, Age: {Age}");
    }
}
class Program
{
    static void Main()
    {
        // Using default constructor
        Person person1 = new Person();
        person1.DisplayInfo();
        // Using parameterized constructor
        Person person2 = new Person("Ravi", 10);
        person2.DisplayInfo();
    }
}
```

OUTPUT



```
● shivam@fedora:~/Code/C#/C#assignment$ dotnet run
Name: Unknown, Age: 0
Name: Ravi, Age: 10
```

5. Develop a C# program to create a class named shape. Create three sub classes namely: circle, triangle and square, each class has two member functions named draw () and erase (). Demonstrate polymorphism concepts by developing suitable methods, defining member data and main program.

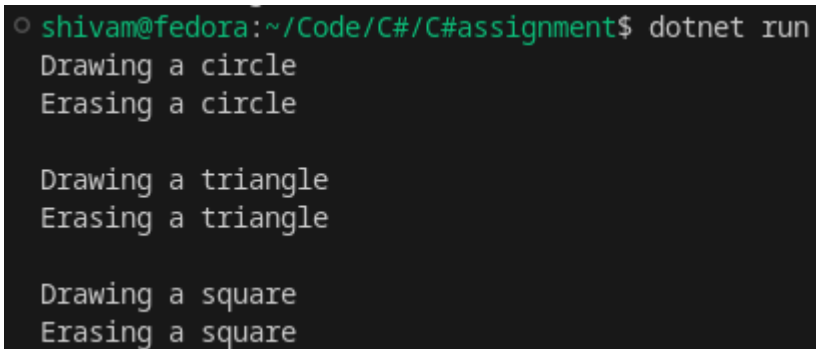
```
using System;
// Base class Shape
class Shape
{
    public virtual void Draw()
    {
        Console.WriteLine("Drawing a shape");
    }
    public virtual void Erase()
    {
        Console.WriteLine("Erasing a shape");
    }
}
// Subclass Circle
class Circle : Shape
{
    public override void Draw()
    {
        Console.WriteLine("Drawing a circle");
    }
    public override void Erase()
    {
        Console.WriteLine("Erasing a circle");
    }
}
// Subclass Triangle
class Triangle : Shape
{
    public override void Draw()
    {
        Console.WriteLine("Drawing a triangle");
    }
    public override void Erase()
    {
        Console.WriteLine("Erasing a triangle");
    }
}
// Subclass Square
class Square : Shape
{

```

```

    public override void Draw()
    {
        Console.WriteLine("Drawing a square");
    }
    public override void Erase()
    {
        Console.WriteLine("Erasing a square");
    }
}
class Program
{
    static void Main()
    {
        // Creating objects of different shapes
        Shape[] shapes = new Shape[3];
        shapes[0] = new Circle();
        shapes[1] = new Triangle();
        shapes[2] = new Square();
        // Demonstrating polymorphism
        foreach (Shape shape in shapes)
        {
            shape.Draw();
            shape.Erase();
            Console.WriteLine(); // Add a newline for separation
        }
        Console.ReadLine(); // Keep the console window open
    }
}

```



```

shivan@fedora:~/Code/C#/C#assignment$ dotnet run
Drawing a circle
Erasing a circle

Drawing a triangle
Erasing a triangle

Drawing a square
Erasing a square

```

6. Develop a C# program to read a text file and copy the file contents to another text file.

Code :


```

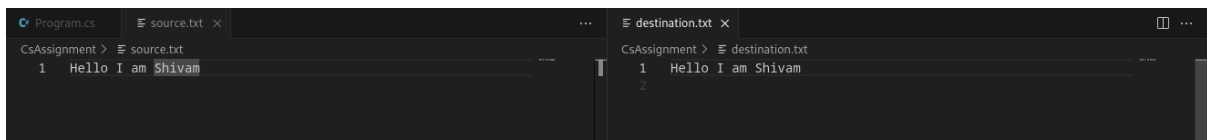
using System;
using System.IO;
class Program
{
    static void Main()
    {
        string sourceFilePath = "source.txt"; // Path to the source file
        string destinationFilePath = "destination.txt"; // Path to the destination file
        try
        {
            // Read the contents of the source file
            string[] lines = File.ReadAllLines(sourceFilePath);
            // Write the contents to the destination file
            File.WriteAllLines(destinationFilePath, lines);
            Console.WriteLine("File contents copied successfully.");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"An error occurred: {ex.Message}");
        }
        Console.ReadLine(); // Keep the console window open
    }
}

```

```

shivam@fedora:~/Code/C#/CsAssignment$ dotnet run
File contents copied successfully.

```



7. Develop a C# Program to Implement Stack with Push and Pop Operations [Hint: Use class, get/set properties, methods for push and pop and main method]

```

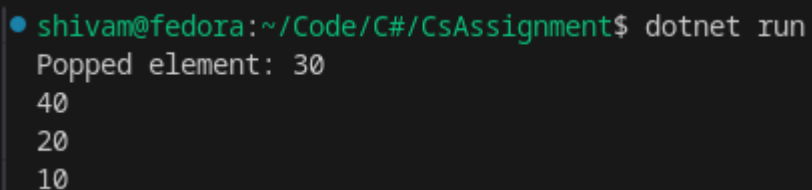
public class Stack<T>
{
    private T[] items;
    private int top = -1;
    public Stack(int capacity)
    {
        items = new T[capacity];
    }
    public bool IsEmpty => top == -1;
    public bool IsFull => top == items.Length - 1;
}

```

```

public void Push(T item)
{
    if (IsFull)
    {
        throw new Exception("Stack overflow");
    }
    items[++top] = item;
}
public T Pop()
{
    if (IsEmpty)
    {
        throw new Exception("Stack underflow");
    }
    return items[top--];
}
}
public class Program
{
    public static void Main(string[] args)
    {
        Stack<int> stack = new Stack<int>(10); // Create a stack with capacity 10
        stack.Push(10);
        stack.Push(20);
        stack.Push(30);
        Console.WriteLine("Popped element: {0}", stack.Pop()); // Print popped element
        stack.Push(40);
        while (!stack.IsEmpty)
        {
            Console.WriteLine(stack.Pop()); // Pop and print all remaining elements
        }
    }
}

```



```

shivam@fedora:~/Code/C#/CsAssignment$ dotnet run
Popped element: 30
40
20
10

```

8. Design a class “Complex” with data members, constructor and method for overloading a binary operator ‘+’. Develop a C# program to read Two complex number and Print the results of addition.

```

class Complex
{
    public double Real { get; set; }
    public double Imaginary { get; set; }
    public Complex(double real, double imaginary)
    {
        Real = real;
        Imaginary = imaginary;
    }
    public static Complex operator +(Complex c1, Complex c2)
    {
        return new Complex(c1.Real + c2.Real, c1.Imaginary + c2.Imaginary);
    }
    public override string ToString()
    {
        return $"{Real} {(Imaginary >= 0 ? "+" : "")}{Imaginary}i";
    }
}
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Enter the first complex number (real imaginary): ");
        double real1 = double.Parse(Console.ReadLine());
        double imag1 = double.Parse(Console.ReadLine());
        Console.WriteLine("Enter the second complex number (real imaginary): ");
        double real2 = double.Parse(Console.ReadLine());
        double imag2 = double.Parse(Console.ReadLine());
        Complex complex1 = new Complex(real1, imag1);
        Complex complex2 = new Complex(real2, imag2);
        Complex sum = complex1 + complex2;
        Console.WriteLine($"The sum of the two complex numbers is: {sum}");
    }
}

```

```

/home/shivam/Code/C#/CsAssignment/Program.cs(24,29
ble double.Parse(string s)'. [/home/shivam/Code/C#
/home/shivam/Code/C#/CsAssignment/Program.cs(25,29
ble double.Parse(string s)'. [/home/shivam/Code/C#
/home/shivam/Code/C#/CsAssignment/Program.cs(27,29
ble double.Parse(string s)'. [/home/shivam/Code/C#
/home/shivam/Code/C#/CsAssignment/Program.cs(28,29
ble double.Parse(string s)'. [/home/shivam/Code/C#
Enter the first complex number (real imaginary):
3
2
Enter the second complex number (real imaginary):
4
5
The sum of the two complex numbers is: 7 +7i

```

9. Develop a C# program to create an abstract class Shape with abstract methods calculateArea() and calculatePerimeter(). Create subclasses Circle and Triangle that extend the Shape class and implement the respective methods to calculate the area and perimeter of each shape

```
using System;
// Abstract class Shape
abstract class Shape
{
    public abstract double CalculateArea();
    public abstract double CalculatePerimeter();
}
// Subclass Circle
class Circle : Shape
{
    private double radius;
    // Constructor
    public Circle(double radius)
    {
        this.radius = radius;
    }
    // Method to calculate area of circle
    public override double CalculateArea()
    {
        return Math.PI * radius * radius;
    }
    // Method to calculate perimeter of circle
    public override double CalculatePerimeter()
    {
        return 2 * Math.PI * radius;
    }
}
// Subclass Triangle
class Triangle : Shape
{
    private double side1, side2, side3;
    // Constructor
    public Triangle(double side1, double side2, double side3)
    {
        this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;
    }
    // Method to calculate area of triangle using Heron's formula
    public override double CalculateArea()
```

```

    {
        double s = (side1 + side2 + side3) / 2;
        return Math.Sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }
    // Method to calculate perimeter of triangle
    public override double CalculatePerimeter()
    {
        return side1 + side2 + side3;
    }
}
class Program
{
    static void Main()
    {
        // Create objects of Circle and Triangle
        Circle circle = new Circle(5);
        Triangle triangle = new Triangle(3, 4, 5);
        // Calculate and display area and perimeter of circle
        Console.WriteLine("Circle:");
        Console.WriteLine($"Area: {circle.CalculateArea()}");
        Console.WriteLine($"Perimeter: {circle.CalculatePerimeter()}");
        // Calculate and display area and perimeter of triangle
        Console.WriteLine("\nTriangle:");
        Console.WriteLine($"Area: {triangle.CalculateArea()}");
        Console.WriteLine($"Perimeter: {triangle.CalculatePerimeter()}");
        Console.ReadLine(); // Keep the console window open
    }
}

```

```

○ shivam@fedora:~/Code/C#/CsAssignment$ dotnet run
Circle:
Area: 78.53981633974483
Perimeter: 31.41592653589793

Triangle:
Area: 6
Perimeter: 12

```

10. Develop a C# program to create an interface Resizable with methods `resizeWidth(int width)` and `resizeHeight(int height)` that allow an object to be resized. Create a class Rectangle that implements the Resizable interface and implements the resize methods.

```

public interface Resizable
{
    void ResizeWidth(int width);
    void ResizeHeight(int height);
}
public class Rectangle : Resizable
{
    private int width;
    private int height;
    public Rectangle(int width, int height)
    {
        this.width = width;
        this.height = height;
    }
    public void ResizeWidth(int newWidth)
    {
        if (newWidth > 0)
        {
            width = newWidth;
            Console.WriteLine($"Resized width to {width}");
        }
        else
        {
            Console.WriteLine("Width must be greater than 0.");
        }
    }
    public void ResizeHeight(int newHeight)
    {
        if (newHeight > 0)
        {
            height = newHeight;
            Console.WriteLine($"Resized height to {height}");
        }
        else
        {
            Console.WriteLine("Height must be greater than 0.");
        }
    }
}
public class Program
{
    public static void Main(string[] args)
    {
        Rectangle rectangle = new Rectangle(5, 3);
        rectangle.ResizeWidth(10);
        rectangle.ResizeHeight(7);
    }
}

```

```
• shivam@fedora:~/Code/C#/CsAssignment$ dotnet run  
Resized width to 10  
Resized height to 7
```