**1)Write a C# program to print fibonacci series without using recursion and using recursion**.

using System;

namespace Exercises  
{

class Fibonacci

{

static void Main(string[] args)

{

int n1 = 0, n2 = 1, n3, i, number;

Console.Write("Enter the number of elements:");

number = int.Parse(Console.ReadLine());

Console.Write(n1 + " " + n2 + " ");

for (i = 2; i < number; ++i)

{

n3 = n1 + n2;

Console.Write(n3+" ");

n1 = n2;

n2 = n3;

}

}

}

}

**OUTPUT:**

[image](https://user-images.githubusercontent.com/97940475/156505831-2f26bc2d-66b5-4ad3-a4f6-791d3b1604fd.png)

**2)Write a C# program to check prime number.**

using System;

namespace Exercises  
{

class PrimeNumber

{

static void Main(string[] args)

{

int n, i, m = 0, flag = 0;

Console.Write("Enter the Number to check Prime:");

n = int.Parse(Console.ReadLine());

m = n / 2;

for (i = 2; i <= m; i++)

{

if (n % i == 0)

{

Console.Write("Number is not Prime.");

flag = 1;

break;

}

}

if (flag == 0)

Console.Write("Number is Prime.");

}

}

}

**OUTPUT:**

[image](https://user-images.githubusercontent.com/97940475/156507398-309d6fd1-b81a-4979-874a-c896e0a7dfaa.png)  
[image](https://user-images.githubusercontent.com/97940475/156507488-00d6461c-3950-40fd-ab25-2b6802477859.png)

**3)Write a C# program to check palindrome number.**

using System;

namespace Exercises  
{

class Palindrome

{

static void Main(string[] args)

{

int n,r, sum = 0, temp;

Console.Write("Enter the Number:");

n = int.Parse(Console.ReadLine());

temp = n;

while(n>0)

{

r = n % 10;

sum = (sum \* 10) + r;

n = n / 10;

}

if (temp == sum)

Console.Write("Number is Palindrome.");

else

Console.Write("Number is not Palindrome.");

}

}

}

**OUTPUT:**

[image](https://user-images.githubusercontent.com/97940475/156508153-c7b85057-dc81-4d44-b761-b7b4c901a9e2.png)  
[image](https://user-images.githubusercontent.com/97940475/156508220-20215039-921b-4161-926b-983137ebfdc2.png)

**4)Write a C# program to print factorial of a number.**

using System;

namespace Exercises  
{

class Factorial

{

static void Main(string[] args)

{

int i, fact = 1, number;

Console.Write("Enter any Number:");

number = int.Parse(Console.ReadLine());

for(i=1;i<=number;i++)

{

fact = fact \* i;

}

Console.Write("Factorial of " + number + " is:" + fact);

}

}

}

**OUTPUT:**

[image](https://user-images.githubusercontent.com/97940475/156509481-bd1ba284-e7f1-4414-b00f-62db37bce03c.png)

**5)Write a C# program to check whether the given element is Armstrong or not.**

using System;

namespace Exercises  
{

class Armstrong

{

static void Main(string[] args)

{

int n, r, sum = 0, temp;

Console.Write("Enter the Number=");

n = int.Parse(Console.ReadLine());

temp = n;

while (n > 0)

{

r = n % 10;

sum = sum + (r \* r \* r);

n = n / 10;

}

if (temp == sum)

Console.Write("Armstrong Number");

else

Console.Write("Not Armstrong Number");

}

}

}

**OUTPUT:**

[image](https://user-images.githubusercontent.com/97940475/156510060-2215a164-8abc-4f12-9217-94506a472d57.png)  
[image](https://user-images.githubusercontent.com/97940475/156510196-2f5c2c79-905d-43eb-a7ce-d4a24c6db184.png)

**6)Write a C# program to find the sum of digits.**

using System;

namespace Exercises  
{

class Sum

{

static void Main(string[] args)

{

int n, sum = 0, m;

Console.Write("Enter a number:");

n = int.Parse(Console.ReadLine());

while(n>0)

{

m = n % 10;

sum = sum + m;

n = n / 10;

}

Console.Write("Sum is=" + sum);

}

}

}

**OUTPUT:**

[image](https://user-images.githubusercontent.com/97940475/156510932-61989d3d-a02d-47a2-96a5-0162dab1f62f.png)

**7)Write a C# program to reverse given number.**

using System;

namespace Exercises  
{

class Reverse

{

static void Main(string[] args)

{

int n, reverse = 0, rem;

Console.Write("Enter a number:");

n = int.Parse(Console.ReadLine());

while(n!=0)

{

rem = n % 10;

reverse = reverse\*10 + rem;

n /= 10;

}

Console.Write("Reversed Number:" + reverse);

}

}

}

**OUTPUT:**

[image](https://user-images.githubusercontent.com/97940475/156511532-c8a227e7-2768-4fea-9788-160601ff810c.png)

**8)C# program to print a Binary Triangle.**

using System;

namespace Exercises  
{

class BinaryTriangle

{

static void Main(string[] args)

{

int number, digit = 1;<br>

Console.Write("\nEnter the number of lines:");

number = Convert.ToInt32(Console.ReadLine());

for (int i = 1; i <= number; i++)

{

for (int space = number - i; space > 0; space--)

{

Console.Write(" ");

}

for (int j = 0; j < i; j++)

{

Console.Write(digit + " ");

digit = (digit == 1) ? 0 : 1;

}

Console.Write("\n");

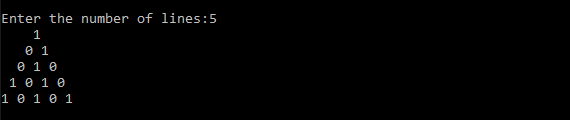
}

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/154415939-8376da3c-f862-42ba-be3e-037a2c9fc26e.png)

**9)C# program to check whether the entered number is an Amicable Number or Not.**

using System;

namespace Exercises  
{

class AmicableNumber

{

static void Main(string[] args)

{

int num1, num2, sum1 = 0, sum2 = 0;

Console.WriteLine("\n-----------AMICABLE NUMBER--------\n");

Console.Write("\nEnter the first number:");

num1 = Convert.ToInt32(Console.ReadLine());

Console.Write("\nEnter the second number:");

num2 = Convert.ToInt32(Console.ReadLine());

for(int i=1;i<num1;i++)

{

if(num1%i==0)

{

sum1 += i;

}

}

for(int i=1;i<num2;i++)

{

if (num2%i==0)

{

sum2 += i;

}

}

if(sum1==num2 && sum2==num1)

{

Console.WriteLine("\nThe numbers {0} and {1} are amicable.", num1, num2);

}

else

{

Console.WriteLine("\nThe numbers {0} and {1} are not amicable.", num1, num2);

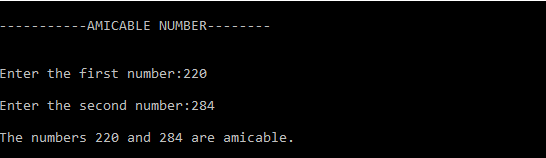
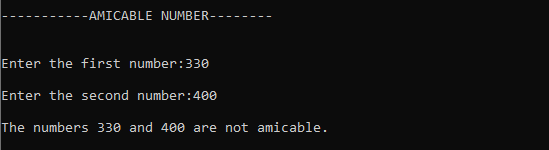
}

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/154425518-d110acb3-9901-4fb7-bd1b-4bed4f611d6c.png)  
[](https://user-images.githubusercontent.com/97940475/154627211-6dc00b0e-3d74-4594-9fb8-8f98fd976e87.png)

**10)C# program to illustrate Multilevel Inheritance with virtual Methods(displaying student details).**

using System;

namespace Exercises  
{

class PersonalDetails

{

string name;

int age;

string gender;

public PersonalDetails(string name, int age, string gender)

{

this.name = name;

this.age = age;

this.gender = gender;

}

public virtual void Display()

{

Console.WriteLine("\n..........PERSONAL DETAILS..........\n");

Console.WriteLine("Name :" + name);

Console.WriteLine("Age :" + age);

Console.WriteLine("Gender :" + gender);

}

}

class CourseDetails : PersonalDetails

{

int regNo;

string course;

int semester;

public CourseDetails(string name, int age, string gender, int regNo, string course, int semester) : base(name, age, gender)

{

this.regNo = regNo;

this.course = course;

this.semester = semester;

}

public override void Display()

{

base.Display();

Console.WriteLine("\n..........COURSE DETAILS..........\n");

Console.WriteLine("Register Number :" + regNo);

Console.WriteLine("Course :" + course);

Console.WriteLine("Semester :" + semester);

}

}

class MarksDetails : CourseDetails

{

int[] marks = new int[5];

int total;

float average;

string grade;

int flagFail;

public MarksDetails(String name, int age, string gender, int regNo, string course, int semester, int[] marks) : base(name, age, gender, regNo, course, semester)

{

total = 0;

for (int i = 0; i < 5; i++)

{

this.marks[i] = marks[i];

total += marks[i];

if (marks[i] < 35)

{

flagFail = 1;

}

}

Calculate();

}

private void Calculate()

{

average = total / 5;

if (flagFail == 1 || average < 40)

grade = "Fail";

else if (average >= 70)

grade = "Distinction";

else if (average >= 60)

grade = "First Class";

else if (average >= 50)

grade = "Second Class";

else

grade = "Pass Class";

}

public override void Display()

{

base.Display();

Console.WriteLine("\n..........MARKS DETAILS..........\n");

Console.Write("Marks in 5 subjects:");

for (int i = 0; i < 5; i++)

Console.Write(marks[i] + " ");

Console.WriteLine();

Console.WriteLine("Total :" + total);

Console.WriteLine("Average :" + average);

Console.WriteLine("Grade :" + grade);

}

}

class MultiLevel

{

public static void Main(string[] args)

{

MarksDetails Student1 = new MarksDetails("Abjhijith", 22, "Male", 20190001, "MCA", 5, new int[] { 77, 80, 98, 95, 90 });

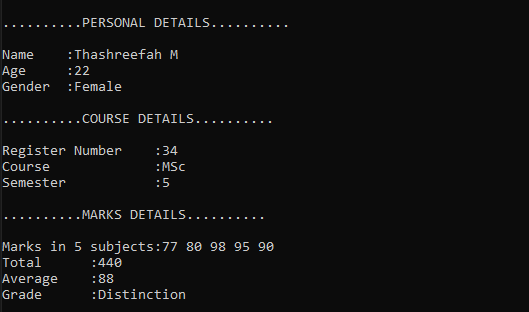
Student1.Display();

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/154632315-63f1fa9d-4728-45d4-b717-fdc4a4cadfb5.png)

**11)C# program to create a grey Code.**

using System;

namespace Exercises  
{

class GrayCode

{

static int getGray(int n)

{

return n ^ (n >> 1);

}

static void Main(string[] args)

{

int InputNum, GrayNum;

Console.Write("\nEnter the decimal number:");

InputNum = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("\nBinary equivalent of {0}:{1}", InputNum, Convert.ToString(InputNum, 2));

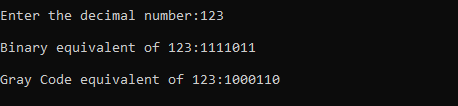
GrayNum = getGray(InputNum);

Console.WriteLine("\nGray Code equivalent of {0}:{1}", InputNum, Convert.ToString(GrayNum, 2));

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940475/154635295-6b3b5acd-63e2-429c-a54c-475eae1e002c.png)

**12)C# program to Calculate volume of 2 boxes and find the resultant volume after addition of 2 boxes by implementing operator overloading**.

using System;

namespace Exercises  
{

class Box

{

float width;

float height;

float length;

public float Volume

{

get { return width \* height \* length; }

}

public Box(float width, float height, float length)

{

this.width = width;

this.height = height;

this.length = length;

}

public static float operator +(Box box1, Box box2)

{

return box1.Volume + box2.Volume;

}

public override string ToString()

{

return "box with width" + width + ",height" + height + "and length" + length;

}

}

class OperatorOverloading

{

public static void Main()

{

Box box1 = new Box(10, 20, 30);

Box box2 = new Box(25, 32, 15);

Console.WriteLine("Volume of {0} is:{1}", box1, box1.Volume);

Console.WriteLine("Volume of {0} is:{1}", box2, box2.Volume);

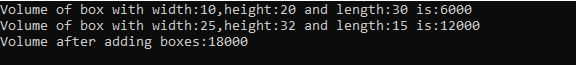
Console.WriteLine("Volume after adding boxes:{0}", box1 + box2);

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/154637878-56fecc75-5864-4c40-b430-9d7d79dcad67.png)

**13)C# program to Implement principles of Delegates (Converting input string to uppercase first,last and entire string)**

using System;

namespace Exercises  
{

class Delegates

{

delegate string UppercaseDelegate(string input);

static string UppercaseFirst(string input)

{

char[] buffer = input.ToCharArray();

buffer[0] = char.ToUpper(buffer[0]);

return new string(buffer);

}

static string UppercaseLast(string input)

{

char[] buffer = input.ToCharArray();

buffer[buffer.Length-1] = char.ToUpper(buffer[buffer.Length-1]);

return new string(buffer);

}

static string UppercaseAll(string input)

{

return input.ToUpper();

}

static void WriteOutput(string input,UppercaseDelegate del)

{

Console.WriteLine("Input String:{0}", input);

Console.WriteLine("Output String:{0}", del(input));

}

static void Main()

{

WriteOutput("tom", new UppercaseDelegate(UppercaseFirst));

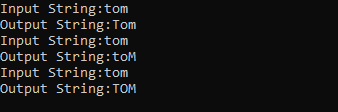
WriteOutput("tom", new UppercaseDelegate(UppercaseLast));

WriteOutput("tom", new UppercaseDelegate(UppercaseAll));

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/156512569-ea5f89ee-833b-4d98-933c-96923a28a2f1.png)

**14)C# program to generate register number automatically for 100 students using static constructor.**

using System;

namespace Exercises  
{

class RegisterNum

{

int regNo;

static int startNum;

static RegisterNum()

{

startNum = 20210000;

}

RegisterNum()

{

regNo = ++startNum;

}

public static void Main(string[] args)

{

for(int i=0;i<100;i++)

{

RegisterNum Student = new RegisterNum();

Console.WriteLine("student {0}:{1}", i + 1, Student.regNo);

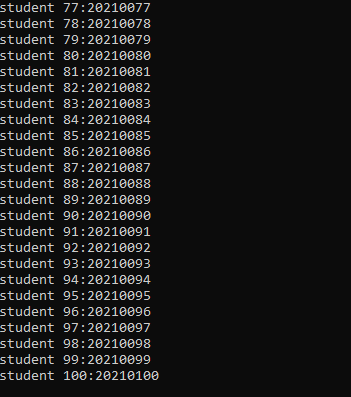
}

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/156513070-8abe00c2-7e65-44f4-92a7-0509dcf28b7a.png)

**15)C# program to find the Frequency of the word "is" in a given sentence.**

using System;

namespace Exercises  
{

class FrequencyIS

{

static void Main(string[] args)

{

int count = 0;

string inputString;

Console.WriteLine("\n-------------Frequency of word 'is'----------");

Console.Write("\nEnter the input string:");

inputString = Console.ReadLine();

char[] separator = { ',', ' ', '.', '!', '\n' };

string testString = inputString.ToLower();

String[] outcomes = testString.Split(separator);

foreach (String s in outcomes)

{

Console.WriteLine(s);

if (s == "is")

count++;

}

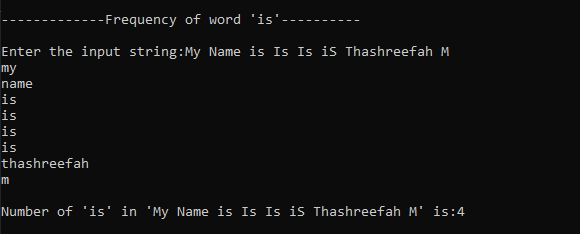
Console.WriteLine("\nNumber of 'is' in '" + inputString + "' is:" + count);

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/156513498-c550351c-484f-4e76-af6c-d44c9fdcc429.png)

**16)C# program that benchmarks 2D,jagged array allocation.**

using System;  
using System.Diagnostics;

namespace Exercises  
{

class BenchmarkAllocation

{

const int \_max = 100000;

static void Main(string[] args)

{

var Arr2D = new int[100, 100];

var ArrJagged = new int[100][];

for(int i=0;i<100;i++)

{

ArrJagged[i] = new int[100];

}

var Stopwatch2D = Stopwatch.StartNew();

for (int i = 0; i < \_max; i++)

{

for (int j = 0; j < 100; j++)

{

for (int k = 0; k < 100; k++)

{

Arr2D[j, k] = k;

}

}

}

Stopwatch2D.Stop();

var StopwatchJagged = Stopwatch.StartNew();

for (int i = 0; i < \_max; i++)

{

for (int j = 0; j < 100; j++)

{

for (int k = 0; k < 100; k++)

{

ArrJagged[j][k] = k;

}

}

}

StopwatchJagged.Stop();

Console.Write("\nTime taken for allocation in case of 2D array:");

Console.WriteLine(Stopwatch2D.Elapsed.TotalMilliseconds+"milliseconds");

Console.Write("\nTime taken for allocation in case of Jagged array:");

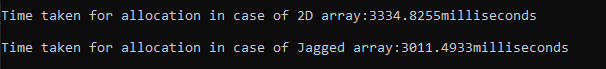
Console.WriteLine(StopwatchJagged.Elapsed.TotalMilliseconds + "milliseconds");

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/156513900-608e4216-5a95-4d71-97aa-bbf4ca0c907b.png)

**17)C# program to find the sum of the values on Diagonal of the Matrix.**

using System;

namespace Exercises  
{

class SumofDiagonals

{

static void Main(string[] args)

{

int MaxRow, MaxCol, Sum = 0;

int[,] Matrix;

Console.WriteLine("\n--------SUM OF DIAGONAL OF MATRIX------\n");

Console.Write("\nEnter the number of rows:");

MaxRow = Convert.ToInt32(Console.ReadLine());

Console.Write("\nEnter the number of columns:");

MaxCol = Convert.ToInt32(Console.ReadLine());

if (MaxRow != MaxCol)

{

Console.WriteLine("\nThe Dimensional entered are not of Square Matrix");

Console.WriteLine("\nExiting the Program");

return;

}

Matrix = new int[MaxRow, MaxCol];

for (int i = 0; i < MaxRow; i++)

{

for (int j = 0; j < MaxCol; j++)

{

Console.Write("\nEnter the ({0},{1})th element of the matrix:", (i + 1), (j + 1));

Matrix[i, j] = Convert.ToInt32(Console.ReadLine());

}

}

Console.WriteLine("\nThe entered Matrix is:");

for (int i = 0; i < MaxRow; i++)

{

for (int j = 0; j < MaxCol; j++)

{

Console.Write(" " + Matrix[i,j]);

if (i==j)

{

Sum += Matrix[i,j];

}

}

Console.WriteLine();

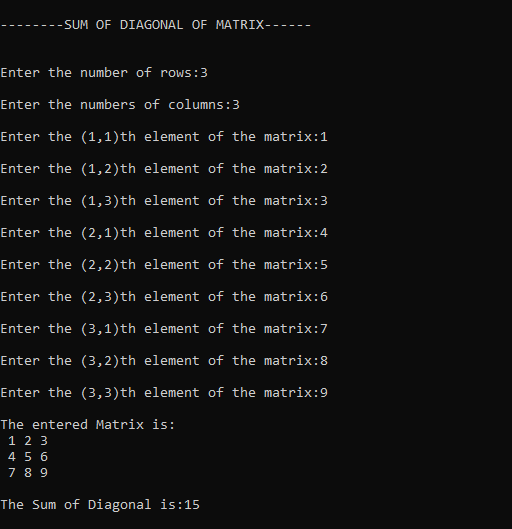
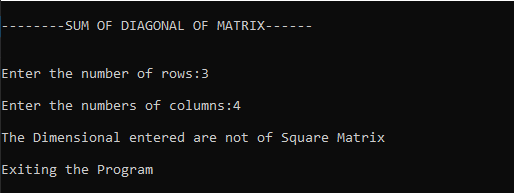
}

Console.WriteLine("\nThe Sum of Diagonal is:" + Sum);

}

}

}

**OUTPUT:**[](https://user-images.githubusercontent.com/97940475/156514596-64b0e219-8326-4969-b852-88756f68f226.png)  
[](https://user-images.githubusercontent.com/97940475/156514701-169831ff-df47-41d0-95eb-9d975e7dc3ca.png)

**18)C# program to create a file,check the existence of a file and read the content of the file.**

using System;  
using System.IO;

namespace Exercises  
{

class FileRead

{

public static void Main()

{

string fileName;

while(true)

{

Console.WriteLine("\n--------MENU--------\n");

Console.WriteLine("\n1.Create a File");

Console.WriteLine("\n2.Existence of the File");

Console.WriteLine("\n3.Read the content of the File");

Console.WriteLine("\n4.Exit");

Console.Write("\nEnter your choice:");

int ch = int.Parse(Console.ReadLine());

switch(ch)

{

case 1:

Console.Write("\nEnter the file name to create:");

fileName = Console.ReadLine();

Console.Write("\nWrite the content of the file:\n");

string r= Console.ReadLine();

using (StreamWriter fileStr = File.CreateText(fileName))

{

fileStr.WriteLine(r);

}

Console.WriteLine("File is created");

break;

case 2:

Console.Write("\nEnter the file name:");

fileName = Console.ReadLine();

if(File.Exists(fileName))

{

Console.WriteLine("File exists...");

}

else

{

Console.WriteLine("File does not exist in the current directory!");

}

break;

case 3:

Console.Write("Enter the file name to read the contents:\n");

fileName = Console.ReadLine();

if (File.Exists(fileName))

{

using (StreamReader sr = File.OpenText(fileName))

{

string s= " ";

Console.WriteLine("Here is the content of the file:");

while((s = sr.ReadLine()) != null)

{

Console.WriteLine(s);

}

Console.WriteLine(" ");

}

}

else

{

Console.WriteLine("File does not exists");

}

break;

case 4:

Console.WriteLine("\nExisting....");

return;

default:

Console.WriteLine("\nInvalid choice");

break;

}

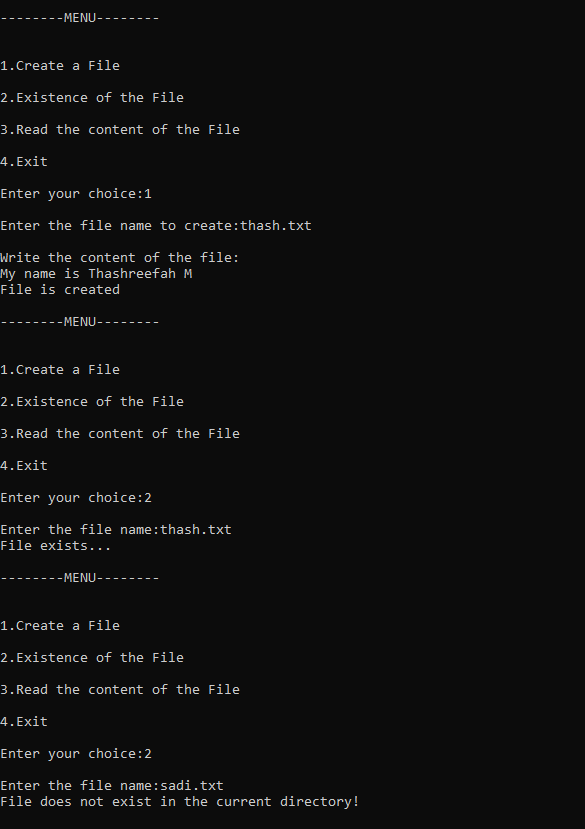
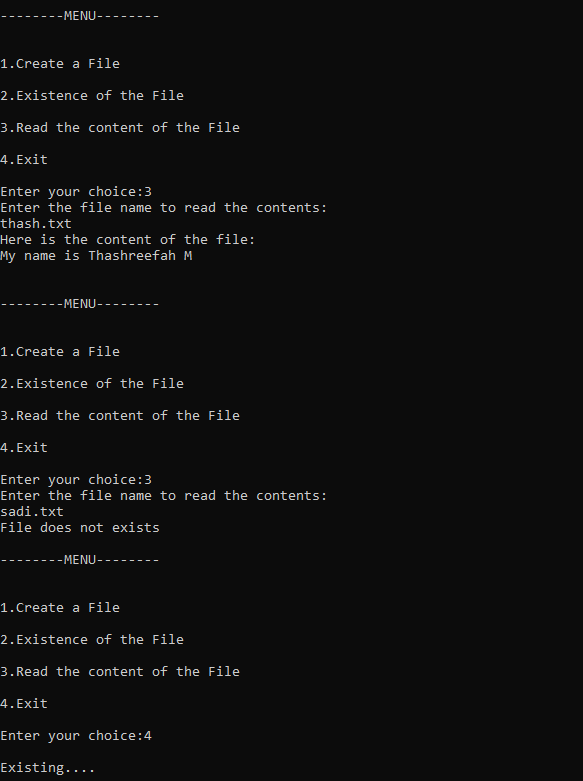
}

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/156515764-1062ccca-7a33-4ba6-b0c9-61bb952f37e1.png) [](https://user-images.githubusercontent.com/97940475/156515810-bd47a052-b090-46ac-ae91-13931c1d11c6.png)

**19)C# program to perform file comparison.**

using System;  
using System.IO;

namespace Exercises  
{

class FileRead1

{

public static void Main()

{

string file1;

string file2;

Console.Write("Enter the first file path:");

file1 = Console.ReadLine();

Console.Write("Enter the second file path:");

file2 = Console.ReadLine();

if(!File.Exists(file1))

{

Console.WriteLine("First file does not exist!");

}

else if (!File.Exists(file2))

{

Console.WriteLine("Second file does not exist!");

}

else if (File.ReadAllText(file1) == File.ReadAllText(file2))

{

Console.WriteLine("Both files contain the same content");

}

else

{

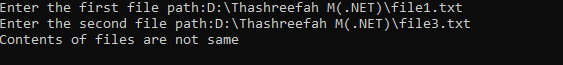
Console.WriteLine("Contents of files are not same");

}

}

}

}

**OUTPUT:**  
[image](https://user-images.githubusercontent.com/97940475/156700946-2a502285-3fae-42c4-9f22-540f197652e8.png)  
[](https://user-images.githubusercontent.com/97940475/156701050-f86e5d4d-0fd7-46b6-bce6-2dafdb0ed42f.png)

**20)C# program to Implement IComparable Interface.**

using System;

namespace Exercises  
{

class Fraction : IComparable

{

int z, n;

public Fraction(int z, int n)

{

this.z = z;

this.n = n;

}

public static Fraction operator +(Fraction a, Fraction b)

{

return new Fraction(a.z \* b.n + a.n \* b.z, a.n \* b.n);

}

public static Fraction operator \*(Fraction a, Fraction b)

{

return new Fraction(a.z \* b.z, a.n \* b.n);

}

public int CompareTo(object obj)

{

Fraction f = (Fraction)obj;

if ((float)z / n < (float)f.z / f.n)

return -1;

else if ((float)z / n > (float)f.z / f.n)

return 1;

else

return 0;

}

public override string ToString()

{

return z + "/" + n;

}

}

class ICompInterface

{

public static void Main()

{

Fraction[] a =

{

new Fraction(5,2),

new Fraction(29,6),

new Fraction(4,5),

new Fraction(10,8),

new Fraction(34,7),

};

Array.Sort(a);

Console.WriteLine("Implementing the IComparable Interface in " + "Displaying Fractions:");

foreach (Fraction f in a)

{

Console.WriteLine(f + " ");

}

Console.WriteLine();

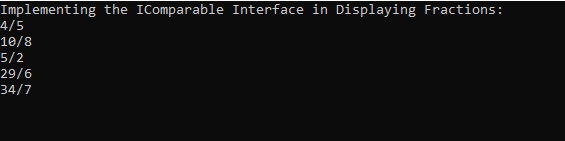
Console.ReadLine();

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/156701481-fea26415-3fd7-4ed3-8ea0-9e035cf69aac.png)

**21)C# program to create Thread Pools.**

using System;  
using System.Threading;

namespace Exercises  
{

class ThreadPoolProg

{

public void ThreadFun1(object obj)

{

int loop = 0;

for (loop = 0; loop <= 4; loop++)

{

Console.WriteLine("Thread1 is executing");

}

}

public void ThreadFun2(object obj)

{

int loop = 0;

for (loop = 0; loop <= 4; loop++)

{

Console.WriteLine("Thread2 is executing");

}

}

public static void Main()

{

ThreadPoolProg TP = new ThreadPoolProg();

for (int i = 0; i < 2; i++)

{

ThreadPool.QueueUserWorkItem(new WaitCallback(TP.ThreadFun1));

ThreadPool.QueueUserWorkItem(new WaitCallback(TP.ThreadFun2));

}

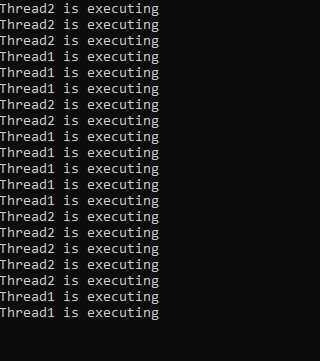
Console.ReadKey();

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/156701877-f1978848-0002-4fb2-8e77-9258d0a26cf3.png)

**22)C# program to demonstrate error handling using Try,Catch and Finally block.**

using System;

namespace Exercises  
{

class ExceptionlHandling

{

static void Main(string[] args)

{

Age a = new Age();

try

{

a.displayAge();

}

catch (AgeIsNegativeException e)

{

Console.WriteLine("AgeIsNegativeException:{0}", e.Message);

}

finally

{

Console.WriteLine("Execution of Finally block is done.");

}

}

}

}

public class AgeIsNegativeException : Exception {

public AgeIsNegativeException(string message) : base(message)

{

}

}

public class Age {

int age = -5;

public void displayAge()

{

if (age < 0)

{

throw (new AgeIsNegativeException("Age cannot be negative"));

}

else

{

Console.WriteLine("Age is:{0}", age);

}

}

}

OUTPUT:  
[image](https://user-images.githubusercontent.com/97940475/156702246-bb6e0595-7c24-4235-bb91-a891ce5349fc.png)

**23)C# Program to Convert digits to words**.

using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;

namespace program1  
{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

label1.Text = NumtoWord(long.Parse(textBox1.Text));

}

public string NumtoWord(long number)

{

string word = "";

if(number==0)

{

return "Zero";

}

if(number<0)

{

return "Minus" + Math.Abs(number);

}

if(number/10000000>0)

{

word += NumtoWord(number / 10000000) + "Crore";

number %= 10000000;

}

if(number/100000>0)

{

word += NumtoWord(number / 100000) + "Lakhs";

number %= 100000;

}

if (number / 1000 > 0)

{

word += NumtoWord(number / 1000) + "Thousand";

number %= 1000;

}

if (number / 100 > 0)

{

word += NumtoWord(number / 100) + "Hundred";

number %= 100;

}

if(number>0)

{

string[] units = new string[] { "Zero", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen" };

string[] Tens = new string[] { "Zero", "Ten", "Twenty", "Thirty", "Fourty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety" };

if(number<20)

{

word += units[number];

}

else

{

word += Tens[number / 10];

if(number%10>0)

{

word += units[number % 10];

}

}

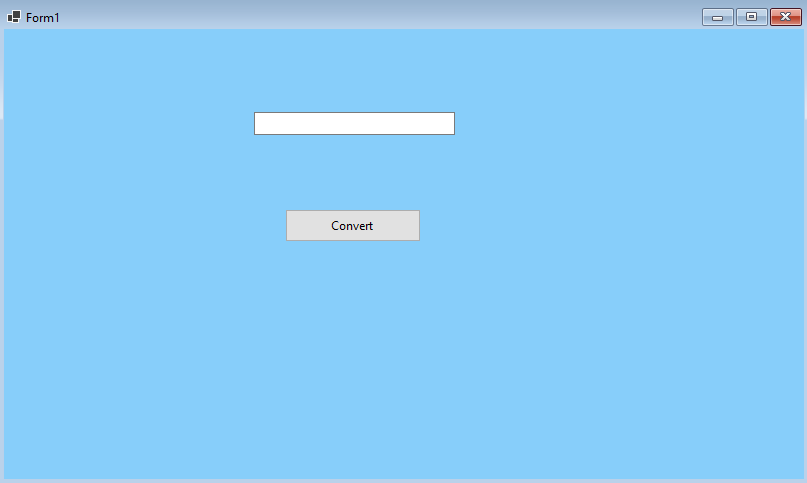
}

return word;

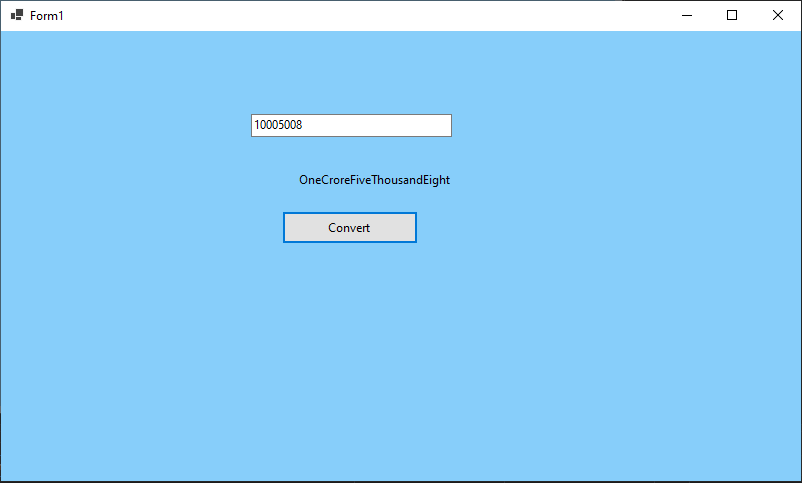
}

}

}

[](https://user-images.githubusercontent.com/97940475/158951740-4428182b-489d-4d11-838b-de0817a6719e.png)

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/158951852-ebd149b8-05af-415f-93e0-a4cc13e81d26.png)

**24)C# program to perform Reversal,Padding and Trimming Operations on string.**

using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;

namespace program2  
{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

string inputString, revstr = "";

int Length;

inputString = textBox1.Text;

Length = inputString.Length - 1;

while (Length >= 0)

{

revstr = revstr + inputString[Length];

Length--;

}

MessageBox.Show("Reverse String Is:" + revstr, "Result");

}

private void button2\_Click(object sender, EventArgs e)

{

string inputString;

inputString = textBox1.Text;

MessageBox.Show("The String After Trimming:" + inputString.Trim(), "Result");

}

private void button3\_Click(object sender, EventArgs e)

{

string inputString;

inputString = textBox1.Text;

inputString = inputString.PadLeft(10, '\*');

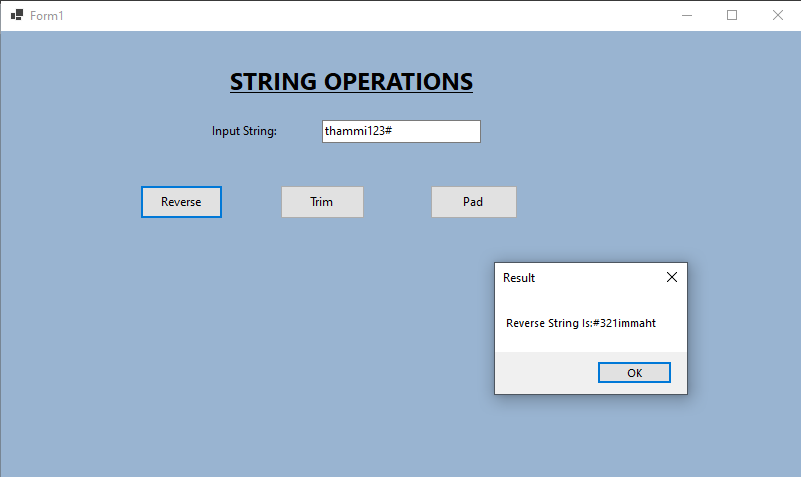
inputString = inputString.PadRight(15, '\*');

MessageBox.Show("String After Padding:" + inputString, "Result");

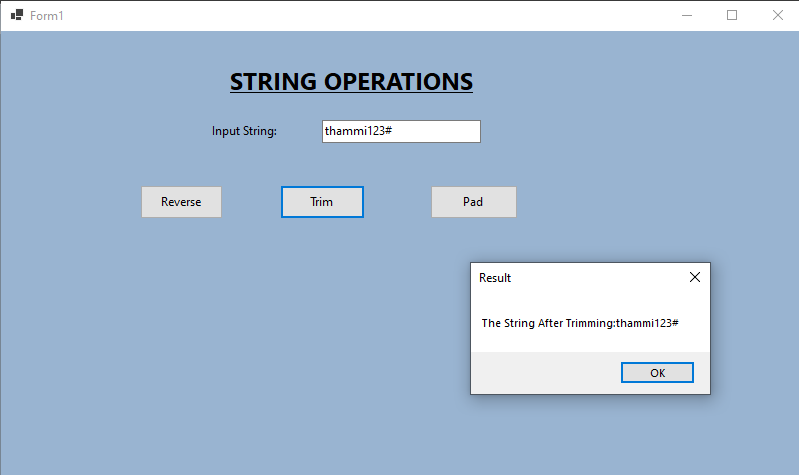
}

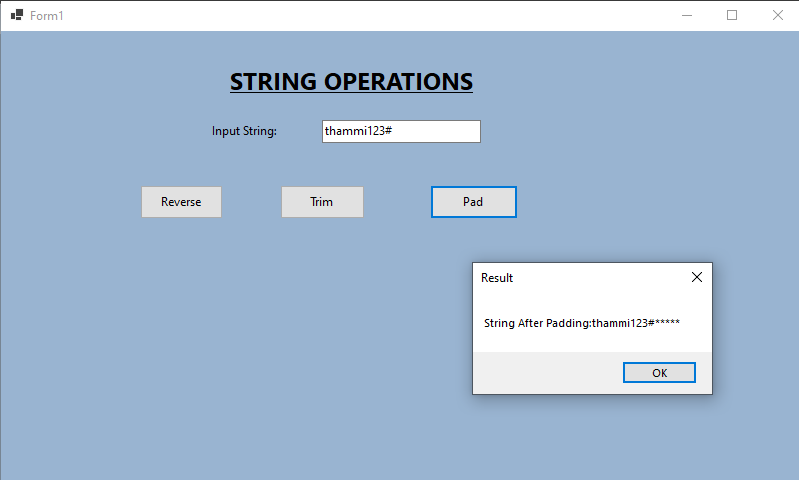
}

}

[](https://user-images.githubusercontent.com/97940475/158953209-ca6466c9-8e48-4097-ba40-22f3a637fbd8.png)[](https://user-images.githubusercontent.com/97940475/158953031-fdcb3f62-bd03-4463-8caa-25a849bc8b5d.png)

**OUTPUT:**

**[](https://user-images.githubusercontent.com/97940475/158953271-07447edc-796e-4761-9412-ada9ba54d9a5.png)**

[](https://user-images.githubusercontent.com/97940475/158953330-1046c6bc-d31c-4732-a886-db218fc187fa.png)

**25)C# Program to create a Progress bar Control.**

using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Threading;  
using System.Windows.Forms;

namespace program4  
{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

backgroundWorker1.WorkerReportsProgress = true;

backgroundWorker1.RunWorkerAsync();

}

private void backgroundWorker1\_DoWork(object sender, DoWorkEventArgs e)

{

for(int i=1;i<=100;i++)

{

Thread.Sleep(50);

backgroundWorker1.ReportProgress(i);

}

}

private void backgroundWorker1\_ProgressChanged(object sender, ProgressChangedEventArgs e)

{

progressBar1.Value = e.ProgressPercentage;

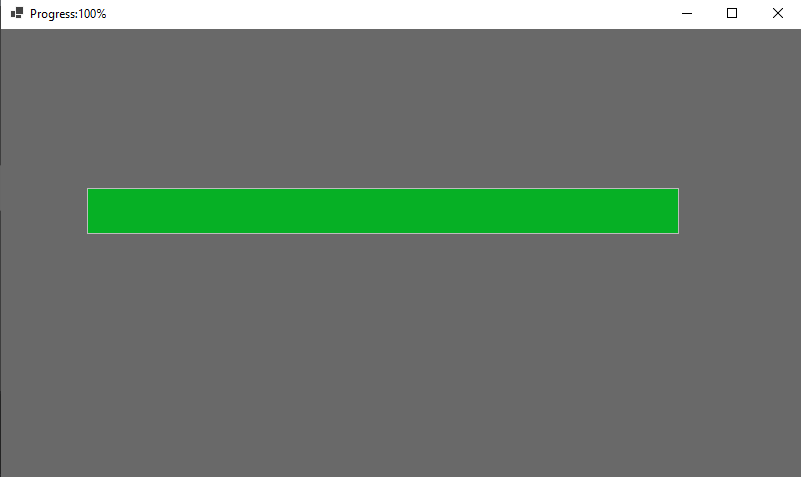
this.Text = "Progress:" + e.ProgressPercentage.ToString() + "%";

}

}

}

[](https://user-images.githubusercontent.com/97940475/158955741-40da118d-35c0-40fd-bcb4-0bb928e04a9c.png)

**[](https://user-images.githubusercontent.com/97940475/158954157-617a042a-46d7-4fdf-80cb-4f030fd3863b.png)OUTPUT:**

**26)Develop a winform application to create Flat Clock.**

using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;

namespace program5 {

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

timer1.Start();

}

private void Form1\_Load(object sender, EventArgs e)

{

System.Timers.Timer timer = new System.Timers.Timer();

timer.Interval = 1000;

timer.Elapsed += Timer\_Elapsed;

timer.Start();

}

private void Timer\_Elapsed(object sender, System.Timers.ElapsedEventArgs e)

{

circularProgressBar1.Invoke((MethodInvoker)delegate

{

circularProgressBar1.Text = DateTime.Now.ToString("hh:mm:ss");

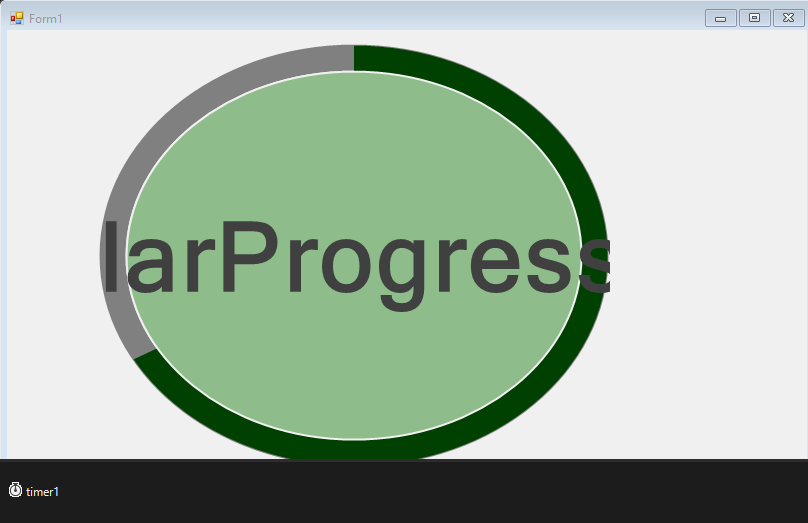
circularProgressBar1.SubscriptText = DateTime.Now.ToString("tt");//AM or PM

});

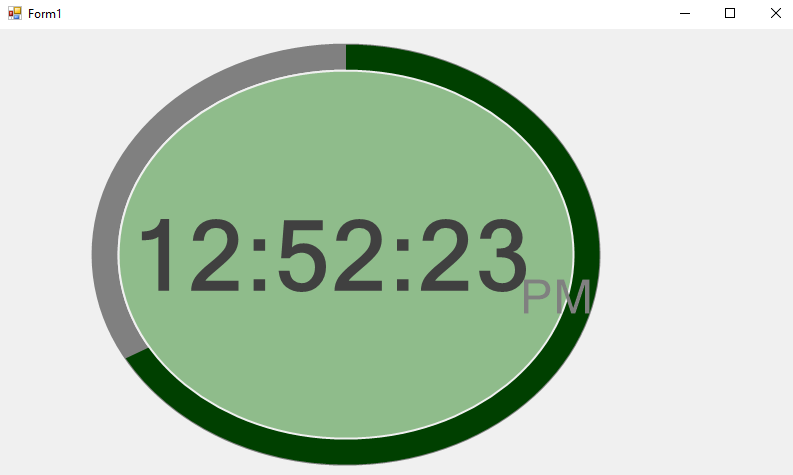
}

}

}

[](https://user-images.githubusercontent.com/97940475/158955367-d5360fd1-944b-470a-86b5-dca7ecf25697.png)

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/158955292-65e86bf7-a286-4151-b931-7cc141f7d9a7.png)

**27)C# Program to perform a number guessing game.**

using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;

namespace program6.\_1  
{

public partial class Form1 : Form

{

static Random r = new Random();

int value;

int guessnum;

int win = 10;

int guess = 1;

Button button1;

TextBox textBox1;

RichTextBox richTextBox1;

RichTextBox richTextBox2;

Label label1;

Label label2;

Label label3;

Label label4;

public Form1()

{

InitializeComponent();

value = r.Next(100);

this.Controls.Clear();

this.BackColor = Color.SkyBlue;

this.AutoSize = true;

this.Padding = new Padding(16);

label1 = new Label();

label1.Text = "Pick a number between 1 to 100";

label1.Bounds = new Rectangle(10, 20, 340, 40);

label1.Font = new Font("Arial", 16);

textBox1 = new TextBox();

textBox1.Bounds = new Rectangle(20, 50, 120, 80);

textBox1.Font = new Font("Arial", 24);

button1 = new Button();

button1.Text = "Check your Guess";

button1.Bounds = new Rectangle(160, 50, 120, 40);

button1.BackColor = Color.LightGray;

button1.Click += new EventHandler(button1\_Click);

label2 = new Label();

label2.Text = "Low Guess";

label2.Bounds = new Rectangle(20, 150, 160, 40);

label2.Font = new Font("Arial", 18);

richTextBox1 = new RichTextBox();

richTextBox1.Bounds = new Rectangle(20, 190, 160, 300);

richTextBox1.Font = new Font("Arial", 16);

label3 = new Label();

label3.Text = "High Guess";

label3.Bounds = new Rectangle(180, 150, 160, 40);

label3.Font = new Font("Arial", 18);

richTextBox2 = new RichTextBox();

richTextBox2.Bounds = new Rectangle(180, 190, 160, 300);

richTextBox2.Font = new Font("Arial", 16);

label4 = new Label();

label4.Bounds = new Rectangle(20, 100, 340, 40);

label4.Font = new Font("Arial", 16);

this.Controls.Add(label1);

this.Controls.Add(textBox1);

this.Controls.Add(button1);

this.Controls.Add(label4);

this.Controls.Add(label2);

this.Controls.Add(label3);

this.Controls.Add(richTextBox1);

this.Controls.Add(richTextBox2);

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox1.Text == "")

{

return;

}

guessnum = Convert.ToInt32(textBox1.Text);

textBox1.Text = String.Empty;

if (win >= 0)

{

if (guessnum == value)

{

MessageBox.Show("You have guessed the number!\nThe number was" + value);

InitializeComponent();

}

else if (guessnum < value)

{

richTextBox1.Text += guessnum + "\n";

MessageBox.Show("Wrong Guess and number of guesses left are" + (10 - guess));

}

else if (guessnum > value)

{

richTextBox2.Text += guessnum + "\n";

MessageBox.Show("Wrong Guess and number of guesses left are" + (10 - guess));

}

guess++;

win--;

}

if (guess == 11)

{

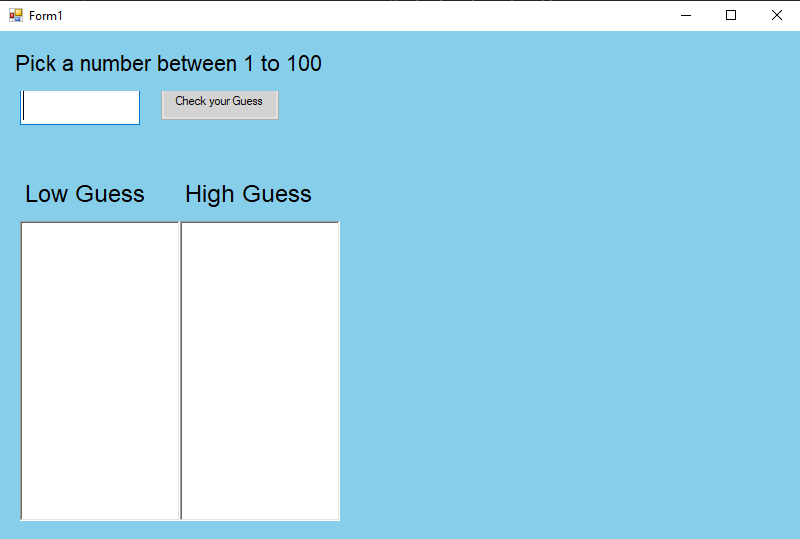
MessageBox.Show("You loose ,Correct Guess is" + value);

}

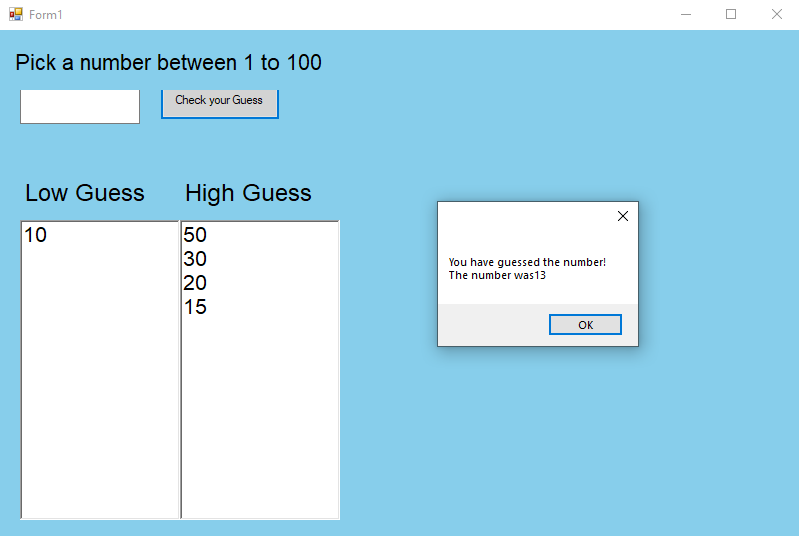
}

}

}

[](https://user-images.githubusercontent.com/97940475/164623341-6ebae466-c785-4ccb-8d33-97c2b227f99f.png)

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/164623755-584cfd17-0e61-4544-8577-53d866b3d2d8.png)

**28)Develop a application to create a notepad.**

using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;

namespace program7  
{

public partial class Form1 : Form

{

private string fileName;

private RichTextBox txtContent;

private ToolBar toolBar;

public Form1()

{

fileName = null;

initializeComponents();

}

void initializeComponents()

{

this.Text = "My notepad";

this.MinimumSize = new Size(600, 450);

this.FormClosing += new FormClosingEventHandler(NotepadClosing);

this.MaximizeBox = true;

toolBar = new ToolBar();

toolBar.Font = new Font("Arial", 16);

toolBar.Padding = new Padding(4);

toolBar.ButtonClick += new ToolBarButtonClickEventHandler(toolBarClicked);

ToolBarButton toolBarButton1 = new ToolBarButton();

ToolBarButton toolBarButton2 = new ToolBarButton();

ToolBarButton toolBarButton3 = new ToolBarButton();

toolBarButton1.Text = "New";

toolBarButton2.Text = "Open";

toolBarButton3.Text = "Save";

toolBar.Buttons.Add(toolBarButton1);

toolBar.Buttons.Add(toolBarButton2);

toolBar.Buttons.Add(toolBarButton3);

txtContent = new RichTextBox();

txtContent.Size = this.ClientSize;

txtContent.Height -= toolBar.Height;

txtContent.Top = toolBar.Height;

txtContent.Anchor = AnchorStyles.Left | AnchorStyles.Right | AnchorStyles.Top | AnchorStyles.Bottom;

txtContent.Font = new Font("Arial", 16);

txtContent.AcceptsTab = true;

txtContent.Padding = new Padding(8);

this.Controls.Add(toolBar);

this.Controls.Add(txtContent);

}

private void toolBarClicked(object sender, ToolBarButtonClickEventArgs e)

{

saveFile();

switch(toolBar.Buttons.IndexOf(e.Button))

{

case 0:

this.Text += "My notepad";

txtContent.Text = string.Empty;

fileName = null;

break;

case 1:

OpenFileDialog openDlg = new OpenFileDialog();

if (DialogResult.OK == openDlg.ShowDialog())

{

fileName = openDlg.FileName;

txtContent.LoadFile(fileName);

this.Text = "My notepad" + fileName;

}

break;

}

}

void saveFile()

{

if (fileName == null)

{

SaveFileDialog saveDlg = new SaveFileDialog();

if (DialogResult.OK == saveDlg.ShowDialog())

{

fileName = saveDlg.FileName;

this.Text += " " + fileName;

}

}

else

{

txtContent.SaveFile(fileName, RichTextBoxStreamType.RichText);

}

}

private void NotepadClosing(object sender,FormClosingEventArgs e)

{

saveFile();

}

/\*static void Main(string[] args)

{

Application.Run(new Form1());

}\*/

private void Form1\_Load(object sender, EventArgs e)

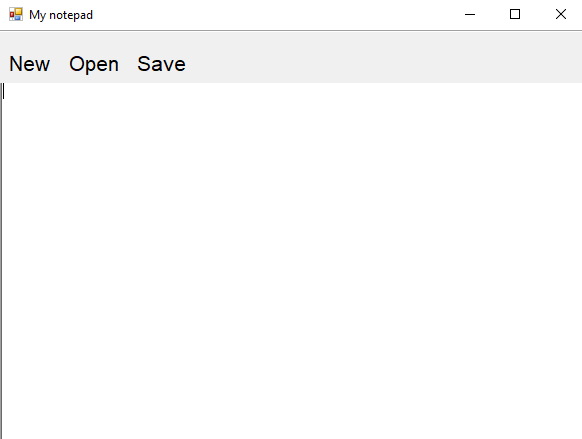
{

}

}

}

**OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/164624781-40f598d6-5532-4fed-b248-50220d2b7678.png)

**29)Develop an application to construct a graphical binary tree where you need to create,add,search and remove nodes.**

using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;  
using System.Drawing.Drawing2D;

namespace program8  
{

public partial class Form1 : Form

{

private Node root;

public Form1()

{

InitializeComponent();

this.root = null;

test();

}

void test()

{

textBox1.Text = "5";

button1\_Click(button1, null);

textBox1.Text = "3";

button1\_Click(button1, null);

textBox1.Text = "2";

button1\_Click(button1, null);

textBox1.Text = "1";

button1\_Click(button1, null);

textBox1.Text = "4";

button1\_Click(button1, null);

textBox1.Text = "7";

button1\_Click(button1, null);

textBox1.Text = "6";

button1\_Click(button1, null);

textBox1.Text = "8";

button1\_Click(button1, null);

}

private void button1\_Click(object sender, EventArgs e)

{

int value = int.Parse(textBox1.Text);

if (root == null)

root = new Node(value);

else

{

if (root.Add(value) == false)

MessageBox.Show("The value already exists!");

}

drawTree();

}

private void button3\_Click(object sender, EventArgs e)

{

root = null;

pictureBox1.Image = null;

}

private void button2\_Click(object sender, EventArgs e)

{

{

int value = int.Parse(textBox1.Text);

if (root != null)

{

bool status = root.Remove(value, root, ref root);

if (status == false)

{

MessageBox.Show("the value does not exists");

}

}

drawTree();

}

}

private void button4\_Click(object sender, EventArgs e)

{

string msg;

int value = int.Parse(textBox1.Text);

if (root == null)

{

msg = "Tree is empty";

}

else

{

if (root.Exists(value))

{

msg = "Value found";

}

else

{

msg = "Value not found";

}

}

MessageBox.Show(msg);

}

void drawTree()

{

if (root != null)

pictureBox1.Image = root.Draw();

else

pictureBox1.Image = null;

this.Update();

}

// static void Main()

// {

// Application.Run(new Form1());

//}

}

class Node

{

internal Node left { get; set; }

internal Node right { get; set; }

internal int value;

internal int center = 12;

private static Bitmap nodeBg = new Bitmap(30, 25);

private static Font font = new Font("Arial", 14);

internal Node(int value)

{

this.value = value;

}

internal bool Add(int value)

{

Node node = new Node(value);

if (value < this.value)

{

if (this.left == null)

{

this.left = node;

return true;

}

else

return this.left.Add(value);

}

else if (value > this.value)

{

if (this.right == null)

{

this.right = node;

return true;

}

else

return this.right.Add(value);

}

return false;

}

internal bool Remove(int value, Node parent, ref Node root)

{

if (value < this.value)

{

if (left != null)

{

return left.Remove(value, this, ref root);

}

}

else if (value > this.value)

{

if (right != null)

{

return right.Remove(value, this, ref root);

}

}

else if (value == this.value)

{

bool isLeft = (this == parent.left);

if (left == null && right == null)

{

if (root == this)

root = null;

else if (isLeft)

parent.left = null;

else

parent.right = null;

}

else if (right == null)

{

if (isLeft)

parent.left = left;

else

parent.right = left;

if (root == this)

root = left;

}

else

{

if (right.left == null)

{

right.left = left;

if (isLeft)

parent.left = right;

else

parent.right = right;

if (root == this)

root = right;

}

else

{

Node node = right;

while (node.left.left != null)

node = node.left;

Console.WriteLine("Node: " + node.value);

this.value = node.left.value;

Console.WriteLine("here");

node.left = null;

}

}

return true;

}

return false;

}

public Image Draw()

{

Size lSize = new Size(nodeBg.Width / 2, 0);

Size rSize = new Size(nodeBg.Width / 2, 0);

Image lNodeImg = null;

Image rNodeImg = null;

int lCenter = 0, rCenter = 0;

if (this.left != null)

{

lNodeImg = left.Draw();

lSize = lNodeImg.Size;

this.center = lSize.Width;

lCenter = left.center;

}

if (this.right != null)

{

rNodeImg = right.Draw();

rSize = rNodeImg.Size;

rCenter = right.center;

}

int maxHeight = (lSize.Height < rSize.Height) ? rSize.Height : lSize.Height;

if (maxHeight > 0)

maxHeight += 35;

Size resultSize = new Size(lSize.Width + rSize.Width, nodeBg.Size.Height + maxHeight);

Bitmap result = new Bitmap(resultSize.Width, resultSize.Height);

Graphics g = Graphics.FromImage(result);

g.SmoothingMode = SmoothingMode.HighQuality;

g.FillRectangle(Brushes.White, new Rectangle(new Point(0, 0), resultSize));

g.DrawImage(nodeBg, lSize.Width - nodeBg.Width / 2, 0);

string str = "" + value;

g.DrawString(str, font, Brushes.Black, lSize.Width - nodeBg.Width / 2 + 7, nodeBg.Height / 2f - 12);

Pen pen = new Pen(Brushes.Black, 1.2f);

float x1 = center;

float y1 = nodeBg.Height;

float y2 = nodeBg.Height + 35;

float x2 = lCenter;

var h = Math.Abs(y2 - y1);

var w = Math.Abs(x2 - x1);

if (lNodeImg != null)

{

g.DrawImage(lNodeImg, 0, nodeBg.Size.Height + 35);

var points1 = new List<PointF>

{

new PointF(x1, y1),

new PointF(x1 - w/6, y1 + h/3.5f),

new PointF(x2 + w/6, y2 - h/3.5f),

new PointF(x2, y2),

};

g.DrawCurve(pen, points1.ToArray(), 0.5f);

}

if (rNodeImg != null)

{

g.DrawImage(rNodeImg, lSize.Width, nodeBg.Size.Height + 35);

x2 = rCenter + lSize.Width;

w = Math.Abs(x2 - x1);

var points = new List<PointF>

{

new PointF(x1, y1),

new PointF(x1 + w/6, y1 + h/3.5f),

new PointF(x2 - w/6, y2 - h/3.5f),

new PointF(x2, y2)

};

g.DrawCurve(pen, points.ToArray(), 0.5f);

}

return result;

}

public bool Exists(int value)

{

bool res = value == this.value;

if (!res && left != null)

res = left.Exists(value);

if (!res && right != null)

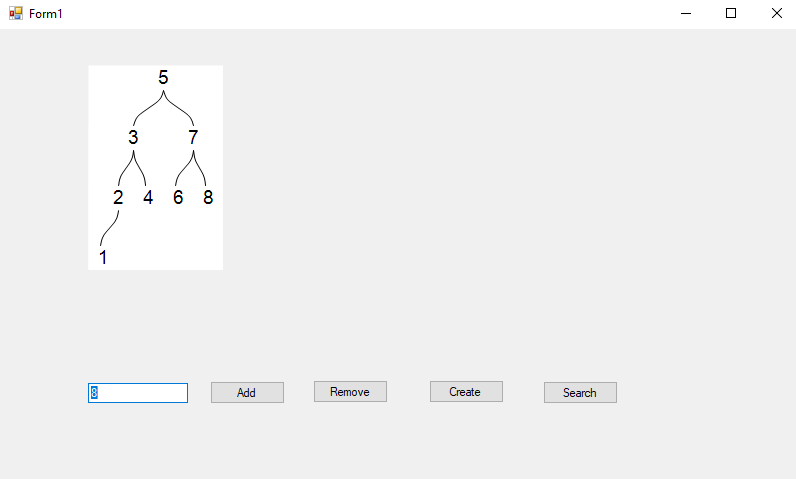
res = right.Exists(value);

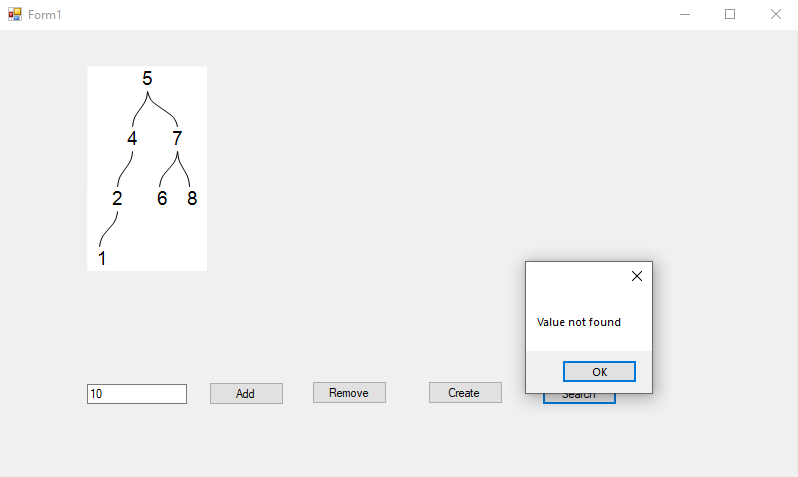
return res;

}

}

}

**[](https://user-images.githubusercontent.com/97940475/164625956-a7e9688f-7b8e-4212-9d33-05f58204d7b6.png)OUTPUT:**

[](https://user-images.githubusercontent.com/97940475/164626077-b1b05eed-b56e-4b89-857b-73c587dbe99b.png)