**1.Write a C# program to print Fibonacci series using Recursion and without using Recursion.**

using System;

namespace Exercises

{

class Fibonacci

{

public 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/97940468/156501447-70eccbc7-b161-4861-8205-830813d9f2ba.png)

**2.Write a C# program to check whether the given number is Prime or not.**

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/97940468/156503073-68c48d68-2eec-44be-a726-a55696457a47.png)  
[image](https://user-images.githubusercontent.com/97940468/156502938-ba88aa3d-2811-400a-a9e5-55286a4abd36.png)

**3.Write a C# program to check whether the given element is Palindrome or not.**

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.WriteLine("Number is Palindrome");

else

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

}

}

}

**OUTPUT:**  
[image](https://user-images.githubusercontent.com/97940468/156503582-7947dee9-9e51-4741-9d6e-f77e1e8e6a28.png)  
[image](https://user-images.githubusercontent.com/97940468/156503708-f9f3b113-87c7-4329-bc82-10c2467bea71.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.WriteLine("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/97940468/156504485-7e1aa166-5edd-4a85-bed7-54e6fc19d1a5.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/97940468/156516873-cec6e9f5-f58c-4260-8fab-92fadf15196c.png)  
[image](https://user-images.githubusercontent.com/97940468/156517059-646b22b5-d1f9-45b7-9213-da925f2245a9.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.WriteLine("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/97940468/156506346-808ed48e-ef28-4f1e-9b87-2c76bc6b75ea.png)

**7.Write a C# program to Reverse a 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/97940468/156506623-7e34867b-2718-4098-849d-e0db5797166b.png)

**8.C# program to print a binary triangle.**

using System;

namespace Exercises

{

class Binarytriangle

{

static void Main(string[] args)

{

int number, digit = 1;

Console.Write("Enter 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;

}

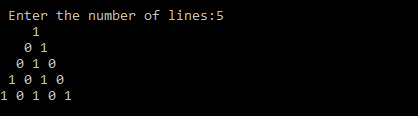
Console.Write("\n");

}

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/154416346-aab9b949-daaf-48d4-b745-2c70aead6524.png)

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

using System;

namespace AmicableNumber

{

class AmicableNumber

{

static void Main(String[] args)

{

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

Console.WriteLine("\n -----AMICABLE NUMBERS-------\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

{

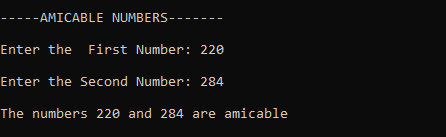
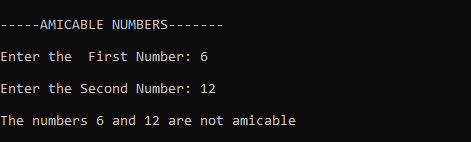
Console.WriteLine("\nThe numbers{0} and {1} are not amicable", num1, num2);

}

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/154634242-01f8d7cc-a01f-4cd7-acc7-284a589c2f88.png)  
[](https://user-images.githubusercontent.com/97940468/154634136-f3c0aad6-1959-489a-8a8f-d82711a909b5.png)

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

using System;

namespace Excercises

{

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 Numbetr:" + 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 = "Firstclass";

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("Toatl:" + total);

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

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

}

}

class Multilevel

{

public static void Main(string[] args)

{

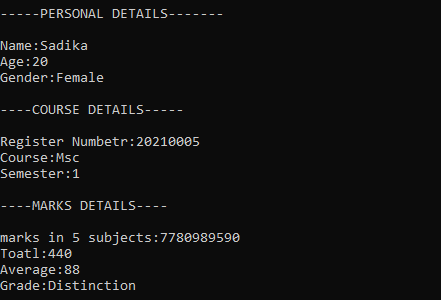
MarksDetails Student1 = new MarksDetails("Sadika", 20, "Female", 20210005, "Msc", 1, new int[] { 77, 80, 98, 95, 90 });

Student1.Display();

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/154634690-af9f0042-05eb-4d39-803f-e4812b40b5b6.png)

**11.C# program to create a Gray 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("\n Enter the decimal number:");

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

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

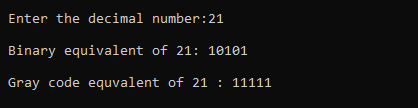
GrayNum = getGray(InputNum);

Console.WriteLine("\n Gray code equvalent of {0} : {1}", InputNum, Convert.ToString(GrayNum, 2));

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/154635339-bd853d51-fab8-4b63-a37a-294b620db775.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 = height;

}

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);

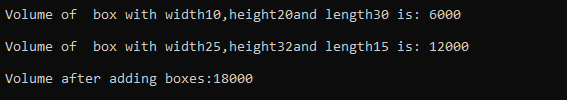
Console.WriteLine("Volume of {0} is: {1}", box2, box2.Volume);

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

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/154636745-5306d598-cfcf-4a34-9ad7-a2719a0f22b5.png)

**13.C# program to implement principle 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));

Console.ReadLine();

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/154637265-daa1bd10-4b40-4727-bbbc-757e2fd09907.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();

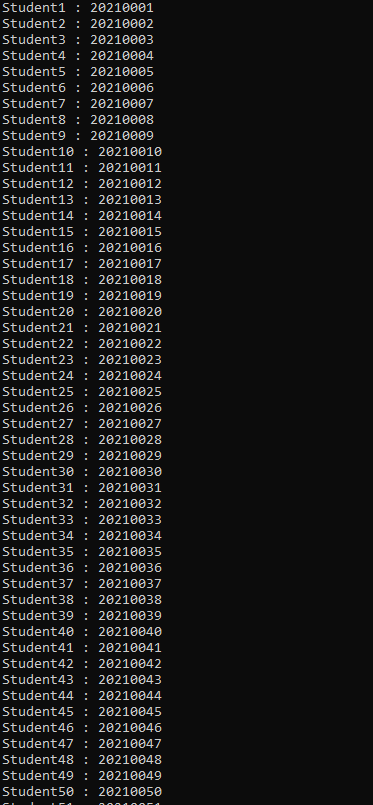
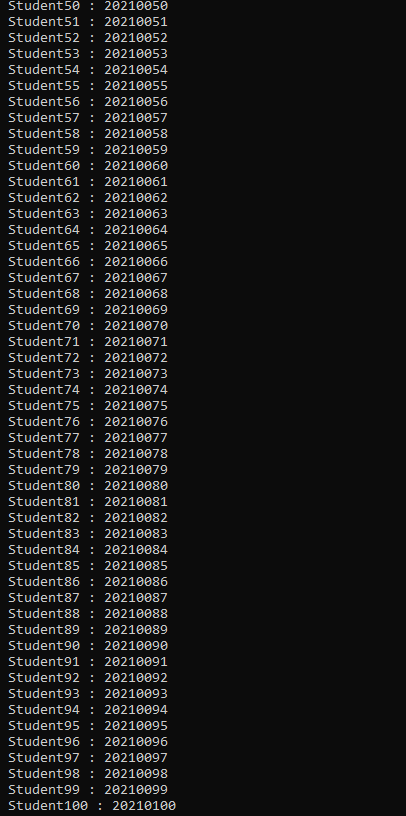
Console.WriteLine("Student{0} : {1}", i + 1, Student.regNo);

}

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/154638059-4c4452f3-c4f9-4042-a085-ac36d5a08166.png) [](https://user-images.githubusercontent.com/97940468/154638265-a58ea4db-40e0-4508-ba48-3650e1c7e9cb.png)

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

using System;

namespace Excercises

{

class FrequencyIS

{

static void Main(string[] args)

{

int count = 0;

string inputString;

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

Console.Write("\n Enter 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++;

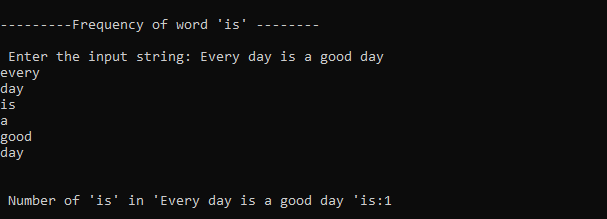
}

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

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/156499718-33cddccc-243e-40c5-bd19-a554c928ba4f.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("\n Time taken for allocation in case of 2D array:");

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

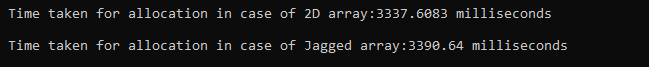
Console.Write("\n Time taken for allocation in case of Jagged array:");

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

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/156507883-5da15a48-adbe-4a1c-bc66-0ce9789049d0.png)

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

using System;

namespace SumofDiagonals

{

class SumofDiagonals

{

static void Main(string[] args)

{

int MaxRow, MaxCol, Sum = 0;

int[,] Matrix;

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

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

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

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

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

if(MaxRow!=MaxCol)

{

Console.WriteLine("\n The Dimensions entered are not of square matrix");

Console.WriteLine("\n Exiting the Program..");

return;

}

Matrix = new int[MaxRow, MaxCol];

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

{

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

{

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

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

}

}

Console.WriteLine("\n The 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();

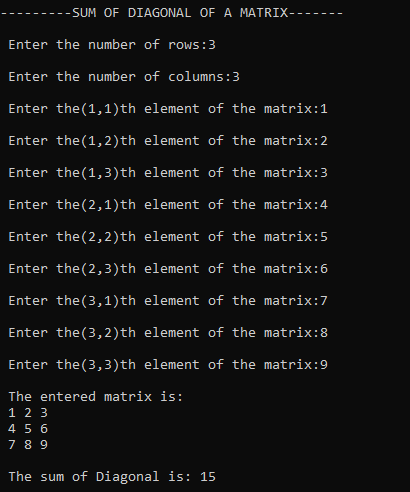
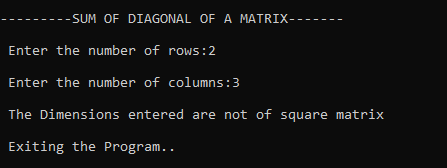
}

Console.WriteLine("\n The sum of Diagonal is" +Sum);

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/156515631-c10ca9c0-a0a1-42f2-9eca-dcd2e4cc149c.png)  
[](https://user-images.githubusercontent.com/97940468/156516425-8b3e22b2-0724-41a8-bad7-2cb17e20eba0.png)

**18.C# program to create a File, check the Existence of a File and Read the Contents 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("\n 1.Create a File");

Console.WriteLine("\n 2. Existence of the File");

Console.WriteLine("\n 3. Read the contents of the File");

Console.WriteLine("\n 4. Exit");

Console.WriteLine("\n Enter your choice:");

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

switch (ch)

{

case 1:

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

fileName = Console.ReadLine();

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

fileName = Console.ReadLine();

if (File.Exists(fileName))

{

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

}

else

{

Console.WriteLine("File doesnot exist in the current directory!");

}

break;

case 3:

Console.Write("\n 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("\n Exiting..");

return;

default:

Console.WriteLine("\n Invalid choice");

break;

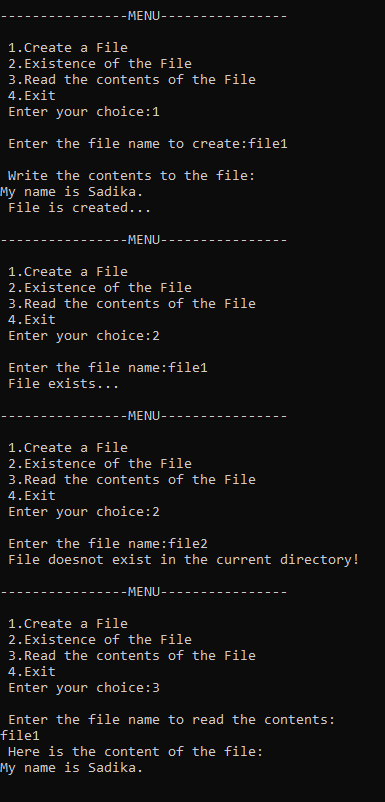
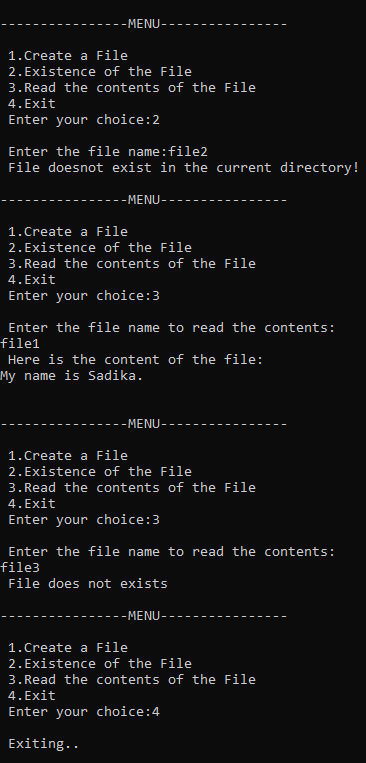
}

}

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/156512591-4cf75486-d9a2-415d-914c-433912745c83.png) [](https://user-images.githubusercontent.com/97940468/156512738-2cfffa40-ba0f-40d7-8cad-63c468246b67.png)

**19.C# program to perform File Comparison.**

using System;

using System.IO;

namespace Exercises

{

class FileComparison

{

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

{

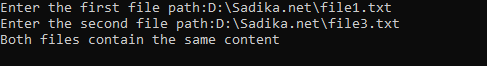
Console.WriteLine("Contents of files are not same");

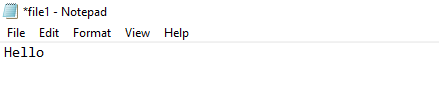
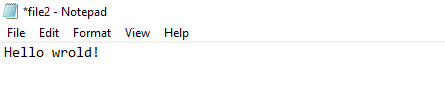
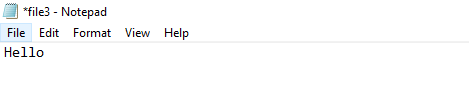
}

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/156513568-cf377b94-7b83-4f69-96d6-558383ca7723.png)  
[image](https://user-images.githubusercontent.com/97940468/156513412-5bfbeb62-31d6-4aab-915a-fd1f1a969031.png)

[](https://user-images.githubusercontent.com/97940468/156513727-920f0b7b-be4e-4de4-8d96-69d411723dd8.png)  
[](https://user-images.githubusercontent.com/97940468/156513843-add3b9f0-4703-498d-b245-2091de36d85a.png)  
[](https://user-images.githubusercontent.com/97940468/156513948-04765c45-365f-44ca-912e-2a6de2c2f03e.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 + " ");

}

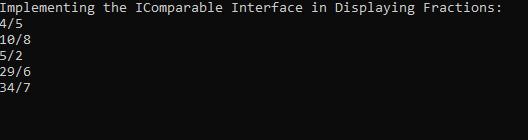
Console.WriteLine();

Console.ReadLine();

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/156514163-3842b7ec-c149-4603-8abd-5cb3bfc2d014.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 excecuting");

}

}

public void ThreadFun2(Object obj)

{

int loop = 0;

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

{

Console.WriteLine("Thread2 is excecuting");

}

}

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));

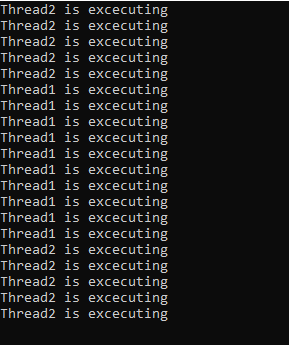
}

Console.ReadKey();

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/156515034-4d6439a4-3d48-4f18-8108-dcea07a7f3e0.png)

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

using System;

namespace Exercises

{

class ExceptionHandling

{

static void Main(string[] args)

{

Age a = new Age();

try

{

a.displayAge();

}

catch(AgeIsNegativeException e)

{

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

}

finally

{

Console.WriteLine("Exception 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/97940468/156514654-4478d3c4-51d0-4c6f-8f5f-7958963316d0.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) + "Lacs";

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", "Ten", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Ninteen" };

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;

}

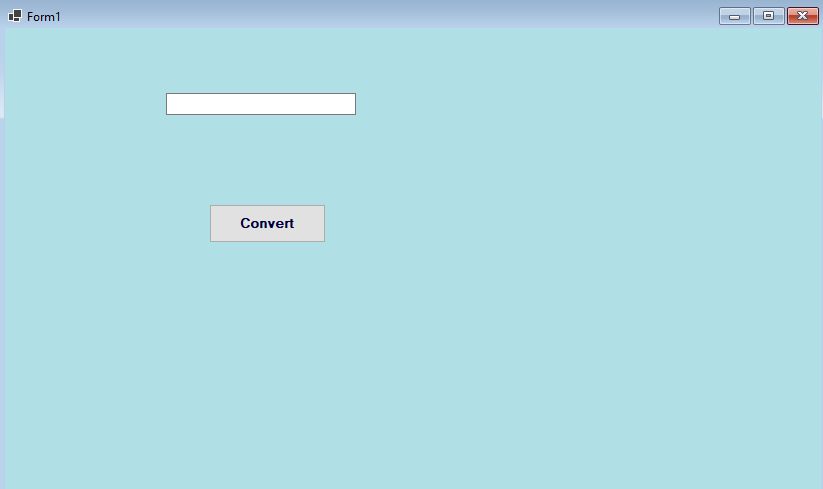
private void Form1\_Load(object sender, EventArgs e)

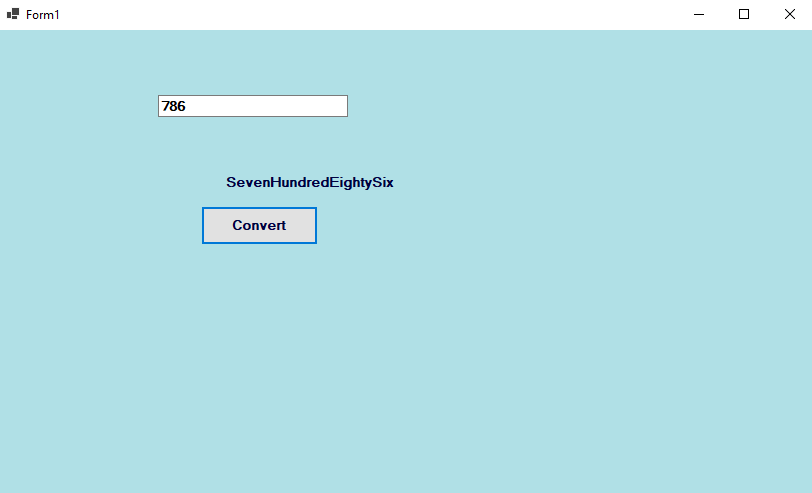
{

}

}

}

**[](https://user-images.githubusercontent.com/97940468/158951983-675cbbd2-78d7-442d-95e2-68b94dca3563.png)OUTPUT:**

[](https://user-images.githubusercontent.com/97940468/158952218-6239cff3-565e-4541-83be-40a50b834229.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 program3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

string inputString, revstr = "";

int Length;

inputString = txtInput.Text;

Length = inputString.Length - 1;

while(Length>=0)

{

revstr = revstr + inputString[Length];

Length--;

}

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

}

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

{

string inputString;

inputString = txtInput.Text;

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

}

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

{

string inputString;

inputString = txtInput.Text;

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

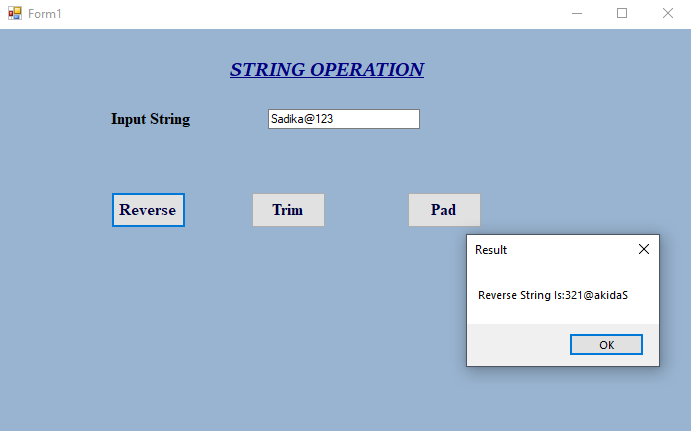
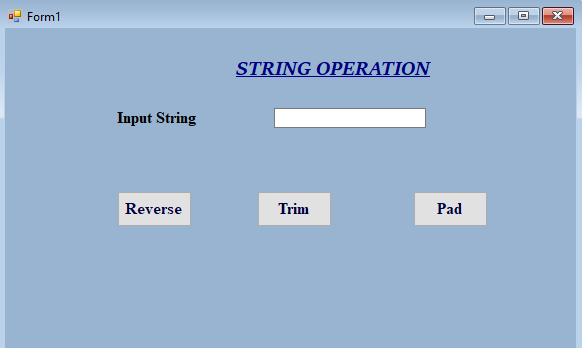
inputString = inputString.PadRight(15, '\*');

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

}

}

}

**[](https://user-images.githubusercontent.com/97940468/158954446-f4bab2b2-760c-4dfb-880e-47d1683f8cec.png)OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/158953700-ceec3471-d49e-4887-9743-a8f9b577cbad.png)

[](https://user-images.githubusercontent.com/97940468/158954652-0cee0826-84ad-4615-b519-ec16e2debb72.png)

[](https://user-images.githubusercontent.com/97940468/158954862-a3dfc3b4-071e-480f-a239-645b8afbd6e2.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;

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)

{

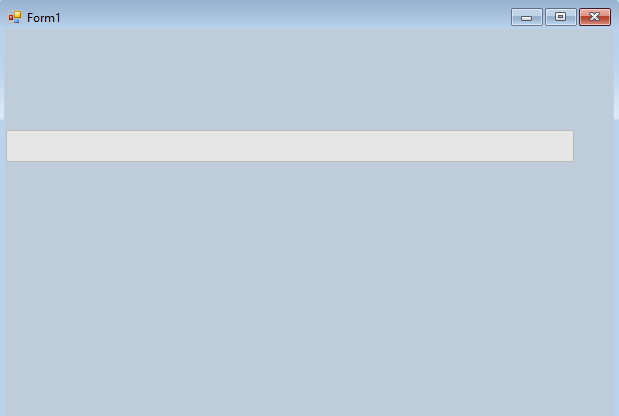
progressBar1.Value = e.ProgressPercentage;

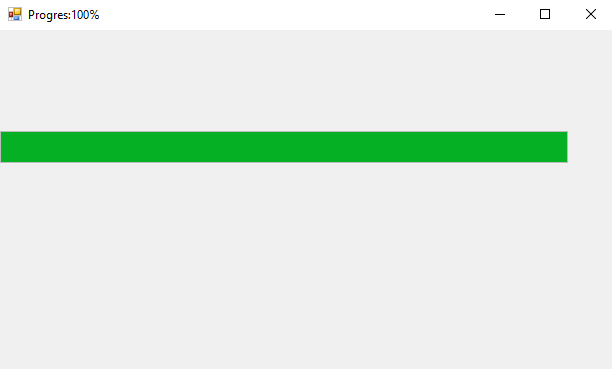
this.Text = "Progres:" + e.ProgressPercentage.ToString() + "%";

}

}

}

**OUTPUT:**  
[](https://user-images.githubusercontent.com/97940468/158955436-1c18f312-13a8-49be-b9d5-a3784ecabfcc.png)

[](https://user-images.githubusercontent.com/97940468/158955706-289e4369-69bf-44eb-8ba1-1f8ef5dec44d.png)

**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();

}

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

{

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

timer.Interval = 100;

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");

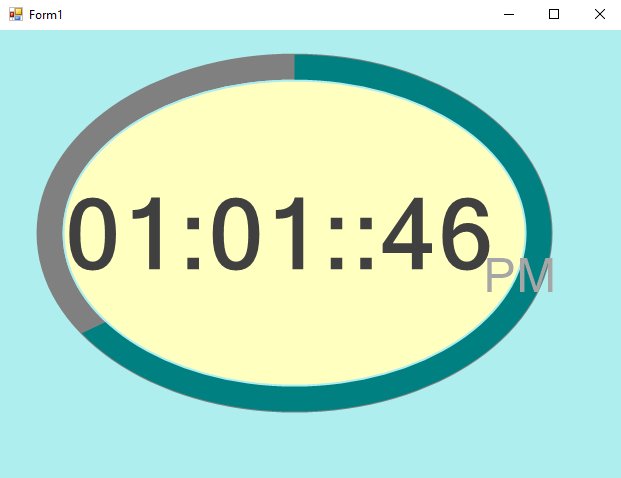
circularProgressBar1.SubscriptText = DateTime.Now.ToString("tt");

});

}

}

}

**[](https://user-images.githubusercontent.com/97940468/158956631-3f38ae8e-8150-49d9-a520-9904a2939379.png)[](https://user-images.githubusercontent.com/97940468/158956123-26bf71fa-0128-458d-b5bd-ba97583d2e87.png)OUTPUT:**  
  
**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 program9

{

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 and 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! \n The 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);

}

}

/\*static void Main()

{

Application.Run(new Form1());

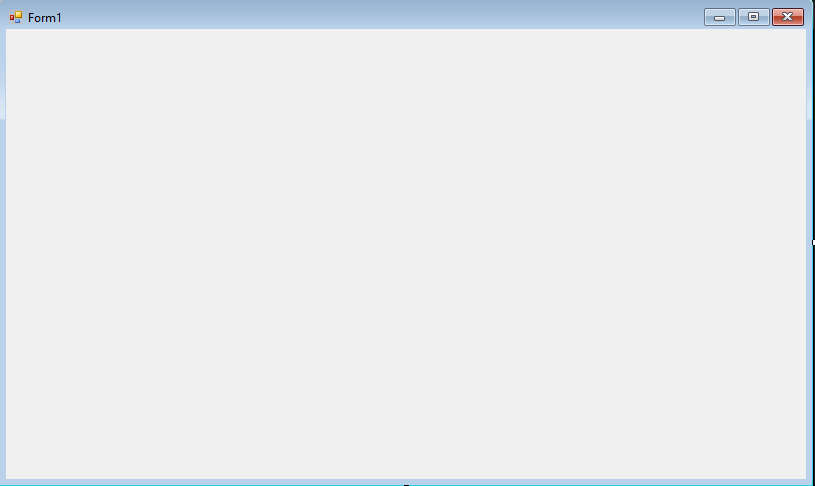
}

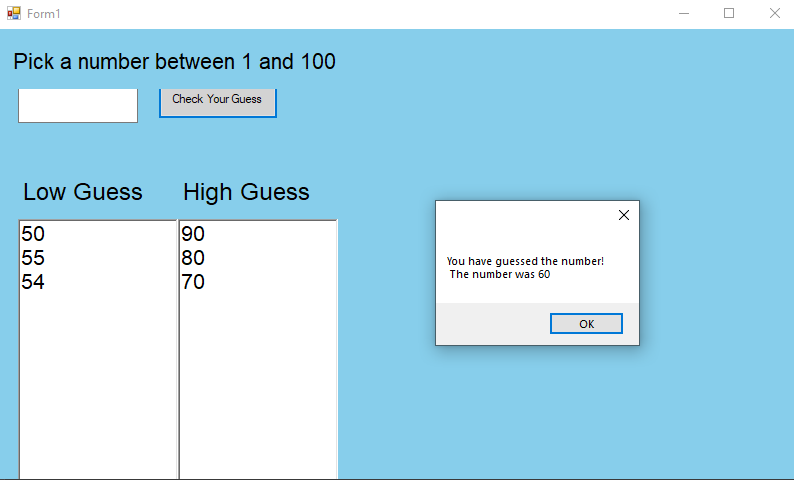
\*/

}

}

**OUTPUT:**

****



**28.Develop an 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 program11

{

public partial class Form1 : Form

{

private string fileName;

private RichTextBox txtContent;

private ToolBar toolBar;

internal 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();

}

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

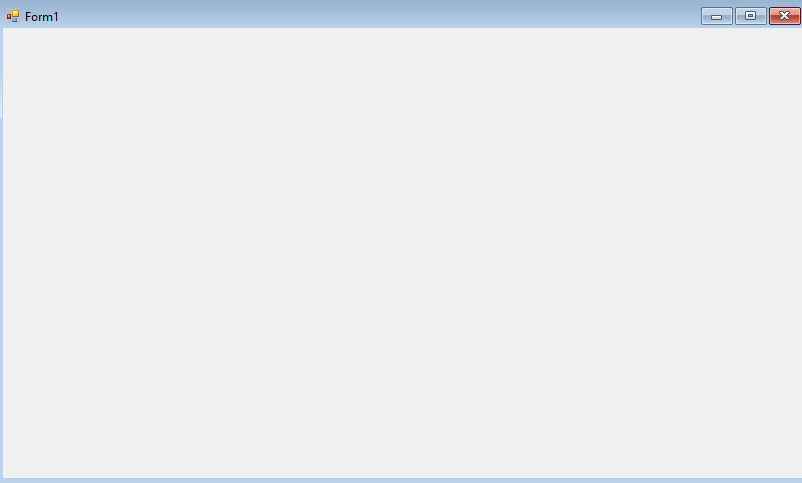
{

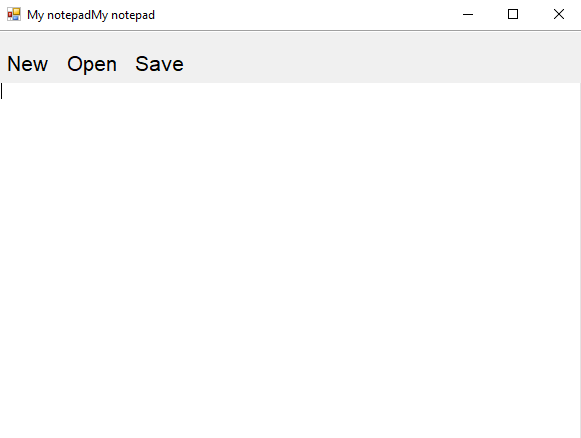
}

}

}

**OUTPUT:**

****

****

**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 program12

{

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 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 button3\_Click(object sender, EventArgs e)

{

root = null;

pictureBox1.Image = null;

}

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();

}

}

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;

}

}

}

**OUTPUT:**

