**LAB 1**

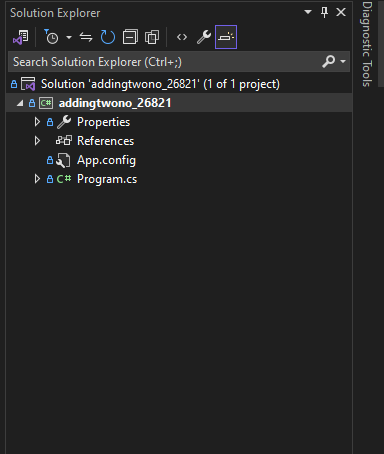
Write a C# program to add two digit using Constructor

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace addingtwono\_26821

{

internal class Program

{

static void Main(string[] args)

{

Console.WriteLine("Enter first number");

int x = Convert.ToInt32 (Console.ReadLine());

Console.WriteLine("Enter the second number");

int y = Convert.ToInt32 (Console.ReadLine());

int z = x + y;

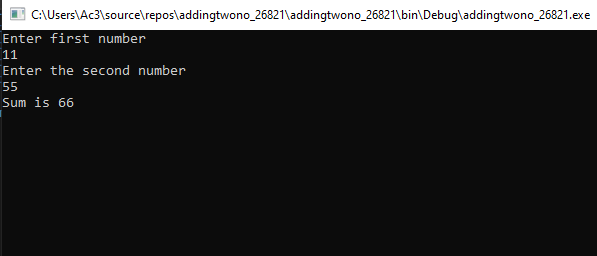
Console.WriteLine("Sum is " + z);

Console.ReadLine();

}

}

}



**LAB 2:**

Write a C# program to initialize and display jagged array elements with sum of each row.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace jagged\_array\_sum\_26821

{

internal class Program

{

static void Main(string[] args)

{

int sum = 0;

int[][] score= new int[2][] { new int[] { 11, 23, 4, 55 }, new int[] { 1, 33, 24, 15, 69 } };

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

{

for (int j = 0; j < score[i].Length; j++)

{

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

sum += Convert.ToInt32(score[i][j]);

}

Console.Write("= " + sum);

sum = 0;

Console.WriteLine();

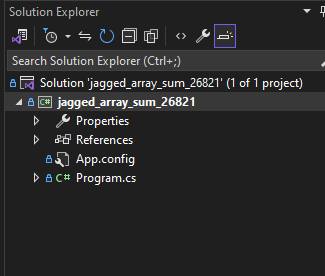
}

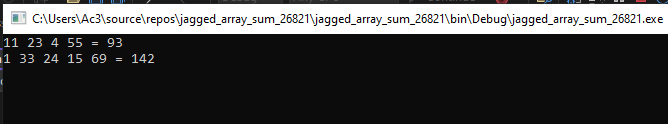
Console.ReadLine();

}

}

}





**LAB 3:**

Write a C# program to initialize and display 2D array elements with sum of each row.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2d\_array\_26821

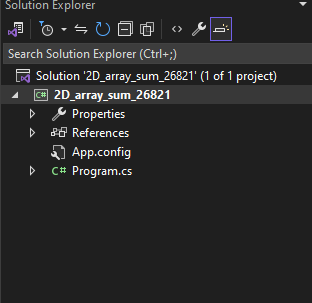
{

internal class Program

{

static void Main(string[] args)

{

 int[][] arr = new int[2][];

arr[0] = new int[] {11,12,3,4,5};

arr[1] = new int[] { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

for (int i =0;i<arr.Length;i++)

{

for (int j = 0; j < arr[i].Length; j++)

{

System.Console.Write(arr[i][j] + " ");

}

Console.WriteLine();

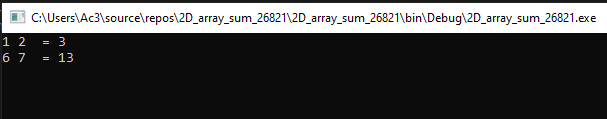
}

Console.ReadLine();

}

}

}



**Lab 4:**

Write a C# Program to calculate Area of Rectangle using Single Inheritance.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace AreaofRectangle\_singleinheritance\_26821

{

internal class Program

{

static void Main(string[] args)

{

Rectangle rect= new Rectangle();

rect.setHeight(10);

rect.setWidth(9);

Console.WriteLine("The area is " + rect.getArea());

Console.ReadLine();

}

}

class Shape

{

protected int width;

protected int height;

public void setWidth(int w)

{

width = w;

}

public void setHeight(int h)

{

height = h;

}

}

class Rectangle: Shape

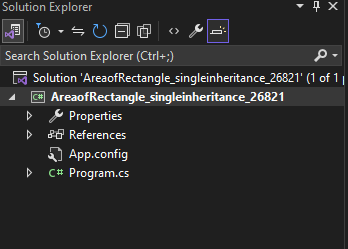
{

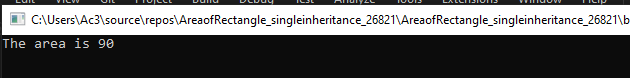
public int getArea()

{

return (width \* height);

}}}





**Lab 5:**

Write a C# program to calculate Area and Paint Cost of Rectangle using multiple Inheritance.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Area\_paintCost\_multipleinheri\_26821

{

internal class Program

{

static void Main(string[] args)

{

Rectangle rect = new Rectangle();

int area;

rect.setHeight(55);

rect.setWidth(11);

area = rect.getArea();

Console.WriteLine("area = " + rect.getArea());

Console.WriteLine("Total Cost is " + rect.getCost(area));

Console.ReadLine();

}

}

class Shape

{

protected int height;

protected int width;

public void setHeight(int h)

{

height = h;

}

public void setWidth(int w)

{

width = w;

}

}

public interface PaintCost

{

int getcost(int area);

}

class Rectangle : Shape, PaintCost

{

public int getArea()

{

return (width \* height);

}

public int getCost(int area)

{

return 90 \* area;

}

public int getcost(int area)

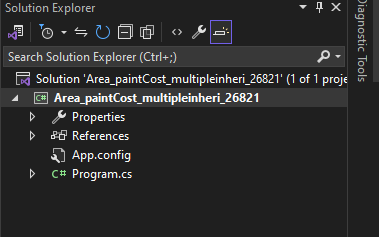
{

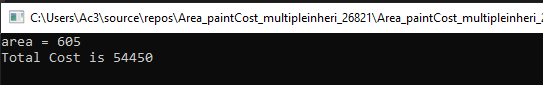
throw new NotImplementedException();

}

}

}





**Lab 6:**

Write a C# program to call base class constructor using “base” Keyword.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.ConstrainedExecution;

using System.Text;

using System.Threading.Tasks;

namespace baseClassConstrutor\_26821

{

internal class Program

{

static void Main(string[] args)

{

car cr = new car();

Console.WriteLine("Car Model: {0}, Speed: {1} ", cr.model, cr.speed);

Console.ReadLine();

}

}

class Vehicle

{

public int speed;

public Vehicle()

{

this.speed = 5;

}

}

class car: Vehicle

{

public string model;

public car() : base()

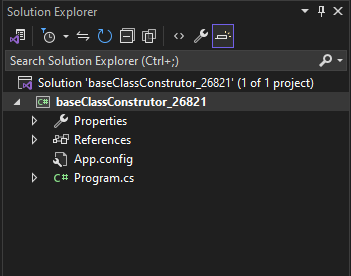
{

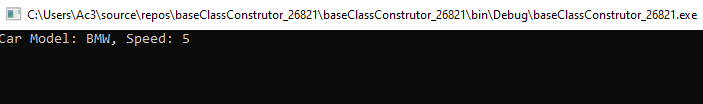
this.model = "BMW";

}

}

}





**Lab 7:**

Create student class with properties for id, name, gender and address.

-Then create a List<Student> to store instances of this class

-the program adds 10 students to the list prints the list

-and then searches for a student by their address using the FindStudentByAddress function.

-finally it prints the result of the search

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace find\_std\_26821

{

internal class Program

{

static void Main(string[] args)

{

List<Student> st = new List<Student>();

st.Add(new Student() {Id=1, Name="Ram", Address="ktm", Gender="Male" });

st.Add(new Student() { Id = 1, Name = "Ram", Address = "ktm", Gender = "Male" });

st.Add(new Student() { Id = 1, Name = "Sam", Address = "ltm", Gender = "Male" });

st.Add(new Student() { Id = 1, Name = "Tam", Address = "ktm", Gender = "female" });

st.Add(new Student() { Id = 1, Name = "Uam", Address = "mtm", Gender = "female" });

st.Add(new Student() { Id = 1, Name = "Vam", Address = "ktm", Gender = "female" });

st.Add(new Student() { Id = 1, Name = "Wam", Address = "ktm", Gender = "Male" });

st.Add(new Student() { Id = 1, Name = "Xam", Address = "ntm", Gender = "Male" });

st.Add(new Student() { Id = 1, Name = "Yam", Address = "ktm", Gender = "Male" });

st.Add(new Student() { Id = 1, Name = "Zam", Address = "otm", Gender = "Male" });

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*List of students\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

foreach (var item in st)

{

Console.WriteLine("Name:{0} Address:{1} Gender:{2}", item.Name, item.Address, item.Gender);

}

Console.WriteLine();

Console.WriteLine("\*\*\*\*\*\*\*\*\*Student with address Ktm\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

List<Student> filterStudent = FindStudentByAddress(st, "ktm");

foreach (var item in filterStudent)

{

Console.WriteLine("Name:{0} Address:{1} Gender:{2}", item.Name, item.Address, item.Gender);

}

Console.ReadLine();

}

public static List<Student> FindStudentByAddress(List<Student> students, string searchAddress)

{

List<Student> filterStudent = new List<Student>();

foreach (Student item in students)

{

if (item.Address == searchAddress)

{

filterStudent.Add(item);

}

}

return filterStudent;

}

}

public class Student

{

public int Id { get; set; }

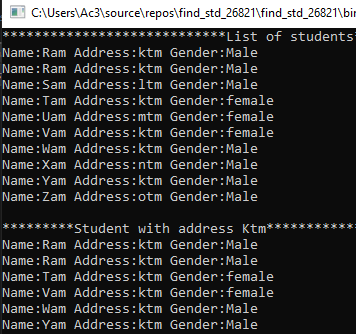
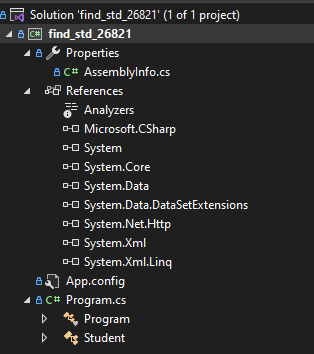
public string Name { get; set; }

public string Address { get; set; }

public String Gender { get; set; }

}

}



**Lab 8:**

Write a C# program to calculate area and paint cost using inheritance use base keyword to

initialize length and breadth.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Area\_PaintCost\_base\_26821

{

internal class Program

{

static void Main(string[] args)

{

Tabletop t = new Tabletop(4,5);

t.Display();

Console.ReadLine();

}

}

class Rectangle

{

//member variables

protected double length;

protected double width;

public Rectangle(double l, double w)

{

length = l;

width = w;

}

public double GetArea()

{

return length \* width;

}

public void Display()

{

Console.WriteLine("Length: {0}", length);

Console.WriteLine("Width: {0}", width);

Console.WriteLine(" Area: {0}", GetArea());

}

}//end class Rectangle

class Tabletop : Rectangle

{

private double cost;

public Tabletop(double l, double w) : base(l, w) { }

public double GetCost()

{

double cost;

cost = GetArea() \* 70;

return cost;

}

public void Display()

{

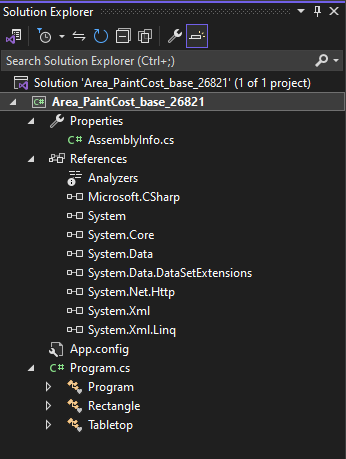
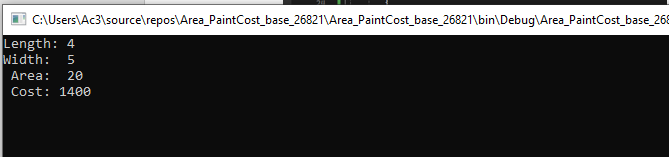
base.Display();

Console.WriteLine(" Cost: {0} ", GetCost());

}

}

}



**Lab 9:**

Write a C# program to illustrate hierarchical inheritance with virtual method.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace hierarchical\_inheritance\_virtual\_method\_26821

{

internal class Program

{

static void Main(string[] args)

{

God gd = new Ram();

God gd2 = new Shiva();

Console.WriteLine(gd.GodName());

Console.WriteLine(gd2.GodName());

Console.ReadLine();

}

}

public class God

{

public virtual string GodName()

{

return "God";

}

}

public class Ram : God

{

public override string GodName()

{

return "Ram";

}

}

public class Shiva: God

{

public override string GodName()

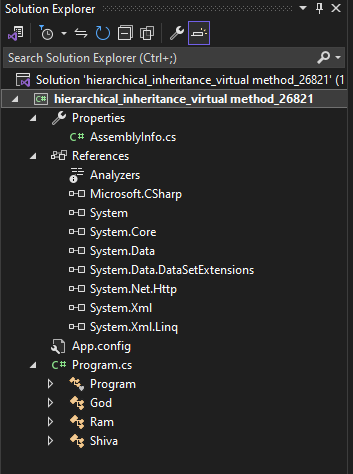
{

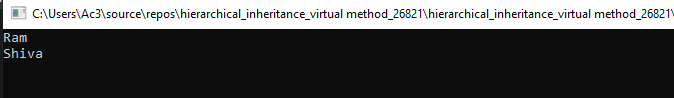
return "Shiva";

}

}

}





**Lab 10:**

WAP to show Polymorphism in C# can be achieved through method overriding.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Polymorphism\_MethodOverriding\_26821

{

internal class Program

{

static void Main(string[] args)

{

God g = new Ram();

God g1 = new Shiva();

Console.WriteLine(g.GodName());

Console.WriteLine(g1.GodName());

Console.ReadLine();

}

}

public class God

{

public virtual string GodName()

{

return " God ";

}

}

public class Ram : God

{

public override string GodName()

{

return " Ram ";

}

}

public class Shiva : God

{

public override string GodName()

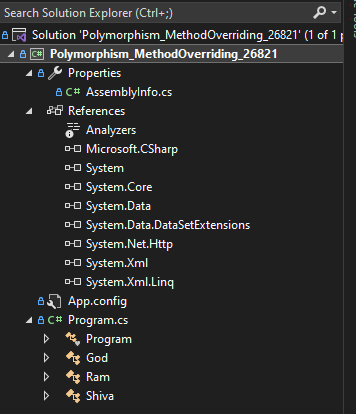
{

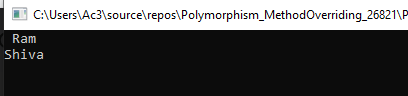
return "Shiva";

}

}

}





**Lab 11:**

WAP to show Polymorphism in C# can be achieved using interfaces.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Polymorphism\_interfaces\_26821

{

internal class Program

{

static void Main(string[] args)

{

IGod circle = new Ram();

circle.GodName();

IGod rectangle = new Shiva();

rectangle.GodName();

Console.ReadLine();

}

}

interface IGod

{

void GodName();

}

class Ram : IGod

{

public void GodName()

{

Console.WriteLine("God Ram");

}

}

class Shiva : IGod

{

public void GodName()

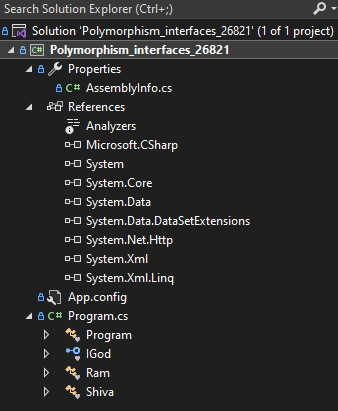
{

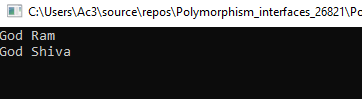
Console.WriteLine("God Shiva");

}

}

}





**Lab 12:**

WAP to show Polymorphism in C# can be achieved using Delegates.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Polymorphism\_Delegates\_26821

{

internal class Program

{

static void Main(string[] args)

{

GodDelegate godMethod;

godMethod = new Ram().GodName;

godMethod();

godMethod = new Shiva().GodName;

godMethod();

Console.ReadLine();

}

}

delegate void GodDelegate();

class Ram

{

public void GodName()

{

Console.WriteLine("God Ram");

}

}

class Shiva

{

public void GodName()

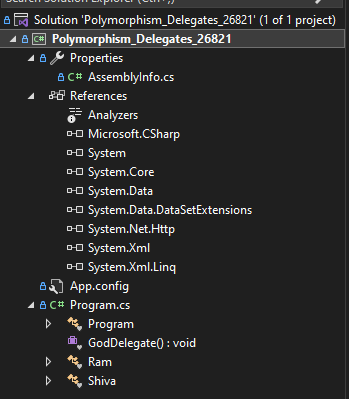
{

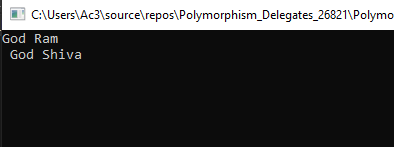
Console.WriteLine(" God Shiva");

}

}

}





**Lab 13:**

Write a C# Sharp program to find the position of a specified word in a given string.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace findPosition\_26821

{

internal class Program

{

static void Main(string[] args)

{

Finder f = new Finder();

string str1 = "The quick brown fox jumps over the lazy dog" ;

Console.WriteLine("Original string:" +str1);

Console.WriteLine(" Position of the word 'fox': " + f.GetResult(str1, "fox"));

Console.WriteLine(" Position of the word 'The': " + f.GetResult(str1, "The"));

Console.WriteLine("Position of the word 'lazy': " + f.GetResult(str1, "lazy"));

Console.ReadLine();

}

}

public class Finder

{

public int GetResult(string text, string word)

{

// Split the input text into words and find the index (position) of the word

// Adding +1 to make (starting from 1 instead of 0)

string[] str = text.Split(' ');

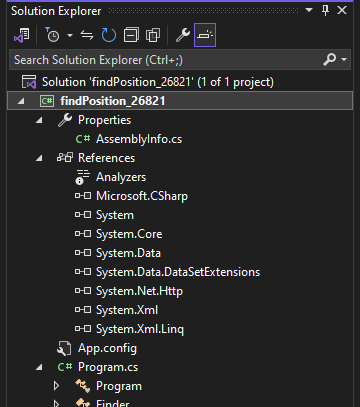
int a = Array.IndexOf(str, word) + 1;

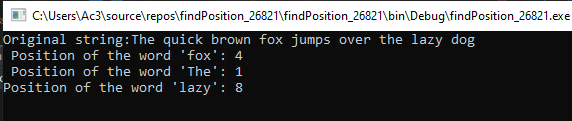
return a;

}

}

}





**Lab 14:**

Write a program in C# Sharp to count the total number of words and Character in a string.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace count\_\_words\_Character\_26821

{

internal class Program

{

static void Main(string[] args)

{

string inputString = "This is a sample string for counting words and characters.";

int wordCount = CountWords(inputString);

int charCount = inputString.Length;

Console.WriteLine("Total number of words: " + wordCount);

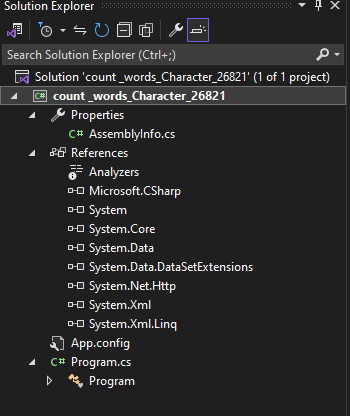
Console.WriteLine("Total number of characters: " + charCount);

Console.ReadLine();

}

private static int CountWords(string input)

{

 string[] words = input.Split(new char[] { ' ', '\t', '\n', '\r' },

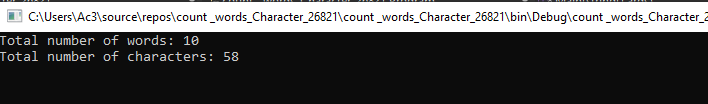
StringSplitOptions.RemoveEmptyEntries);

return words.Length;

}

}

}



**Lab 15:**

Write a program in C# Sharp to count the number of alphabets, digits and special characters in a string.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace count\_alphabets\_digits\_\_specialchar\_26821

{

internal class Program

{

static void Main(string[] args)

{

string inputString = "Hello world 11 ! This is NCC Class ";

// Counting alphabets, digits, and special characters

int alphabetCount = 0;

int digitCount = 0;

int specialCharCount = 0;

foreach (char character in inputString)

{

if (Char.IsLetter(character))

{

alphabetCount++;

}

else if (Char.IsDigit(character))

{

digitCount++;

}

else

{

specialCharCount++;

}

}

Console.WriteLine("Alphabets: " + alphabetCount);

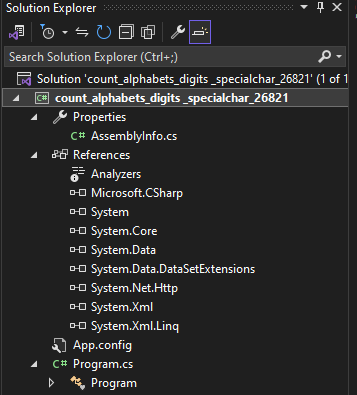
Console.WriteLine("Digits: " + digitCount);

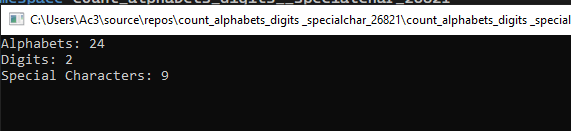
Console.WriteLine("Special Characters: " + specialCharCount);

Console.ReadLine();

}

}}





**Lab 16:**

Write a C# Sharp program to count the number of vowels or consonants in a string.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Count\_Vowel\_Consonant\_26821

{

internal class Program

{

static void Main(string[] args)

{

string inputString = "Hello World Hello universe";

inputString = inputString.ToLower();

int vowelCount = 0;

int consonantCount = 0;

foreach (char character in inputString)

{

if (Char.IsLetter(character))

{

if ("aeiou".Contains(character))

{

vowelCount++;

}

else

{

consonantCount++;

}

}

}

Console.WriteLine("Vowels: " + vowelCount);

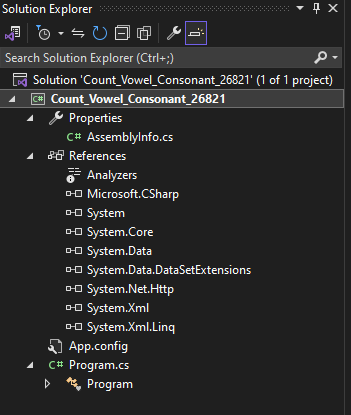
Console.WriteLine("Consonants: " + consonantCount);

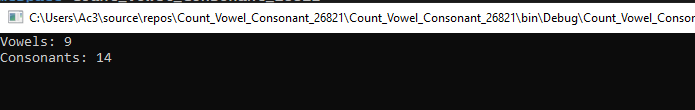
Console.ReadLine();

}

}

}





LAB 17

Write a C# program to raise an ApplicationException using Custom MyException class.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Exception\_26821

{

internal class Program

{

static void Main(string[] args)

{

int d, div, res;

div = Int32.Parse(Console.ReadLine());

d = Int32.Parse(Console.ReadLine());

try

{

if (div == 0)

{

throw new MyException();

}

res = d / div;

Console.WriteLine(" Result: { 0} ", res);

Console.ReadLine();

}

catch (MyException e)

{

e.MyCustomeException();

}

Console.ReadLine();

}

}

class MyException : Exception

{

public void MyCustomeException()

{

Console.WriteLine("Exception occured, divisor should not be zero");

}

}

}

**LAB 18**

Write a C# program to show insert and select student record with given table (tblStudent) with fields (int id, nvarchar(50) name, nvarchar(50) address, nvarchar(50) gender). Also display total no of student from table.

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace DatabaseOperation\_26821

{

internal class Program

{

static void Main(string[] args)

{

StudentInsert\_Select obj = new StudentInsert\_Select();

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

{

Console.WriteLine("Student:" + (i + 1));

Console.Write("Enter Name:");

string name=Console.ReadLine();

Console.Write("Enter address:");

string address = Console.ReadLine();

Console.Write("enter Gender:");

string gender = Console.ReadLine();

obj.Insert(name, address, gender);

Console.WriteLine("Student Inserted");

}

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

Console.WriteLine("Student Records");

DataTable dt= obj.GetAllStudent();

for (int i = 0; i < dt.Rows.Count; i++)

{

Console.WriteLine("Name:{0} Address:{1} Gender:{2}", dt.Rows[i]["Name"].ToString(), dt.Rows[i]["Address"].ToString(), dt.Rows[i]["Gender"].ToString());

}

Console.WriteLine("Total Student" + dt.Rows.Count);

Console.ReadLine();

}

}

public class StudentInsert\_Select

{

public void Insert(string name, string address, string gender)

{

SqlConnection con = new SqlConnection("Data Source=(LocalDB)\\MSSQLLocalDB; Database=SamriddhiData; Integrated Security=true");

SqlCommand cmd = new SqlCommand("insert into tblStudent values (@a,@b,@c)",con);

cmd.Parameters.AddWithValue("@a", name);

cmd.Parameters.AddWithValue("@b", address);

cmd.Parameters.AddWithValue("@c", gender);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

}

public DataTable GetAllStudent()

{

SqlConnection con = new SqlConnection("Data Source=(LocalDB)\\MSSQLLocalDB; Database=SamriddhiData; Integrated Security=true");

SqlCommand cmd = new SqlCommand("select \*from tblStudent", con);

SqlDataAdapter da= new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

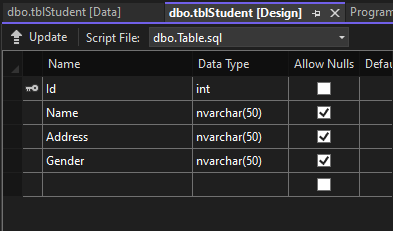
da.Fill(dt);

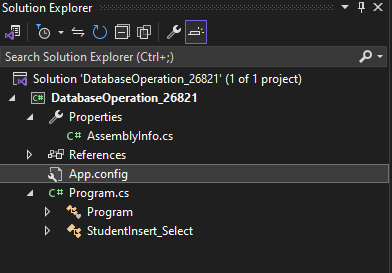
return dt;

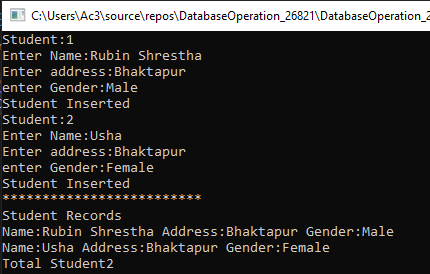
}

}

}







**LAB 19**

Write a C# program to show insert and fetch student record by Gender from given table

(tblStudent) with fields (int id, nvarchar(50) name, int age, nvarchar(50) gender).

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp2

{

internal class Program

{

static void Main(string[] args)

{

StudentInsert\_SelectRecord obj = new StudentInsert\_SelectRecord();

Console.Write("Enter Name:");

string name = Console.ReadLine();

Console.Write("Enter Address:");

string address = Console.ReadLine();

Console.Write("Enter Gender:");

string gender = Console.ReadLine();

obj.Insert(name, address, gender);

Console.WriteLine("Record Inserted");

Console.WriteLine();

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

Console.WriteLine("All Student By Gender");

DataTable dt = obj.GetAllStudentByGender("Male");

for (int i = 0; i < dt.Rows.Count; i++)

{

string n = dt.Rows[i]["Name"].ToString();

string a = dt.Rows[i]["Address"].ToString();

string g = dt.Rows[i]["Gender"].ToString();

Console.WriteLine("Name: {0} Address: {1} Gender: {2}", n, a, g);

}

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

Console.WriteLine("Total No of Student: " + dt.Rows.Count);

Console.ReadLine();

}

}

public class StudentInsert\_SelectRecord

{

public void Insert(string name, string address, string gender)

{

string connStr = @"Data Source=(localdb)\MSSqlLocalDB; Database=SamriddhiDB; Integrated Security=true";

SqlConnection con = new SqlConnection(connStr);

string sql = "insert into tblStudent values(@name,@address,@gender)";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@name", name);

cmd.Parameters.AddWithValue("@address", address);

cmd.Parameters.AddWithValue("@gender", gender);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

}

public DataTable GetAllStudentByGender(string gender)

{

string connStr = @"Data Source=(localdb)\MSSqlLocalDB; Database=SamriddhiDB; Integrated Security=true";

SqlConnection con = new SqlConnection(connStr);

string sql = "select \* from tblStudent where Gender=@gender";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@gender", gender);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();//can hold data in tabular format

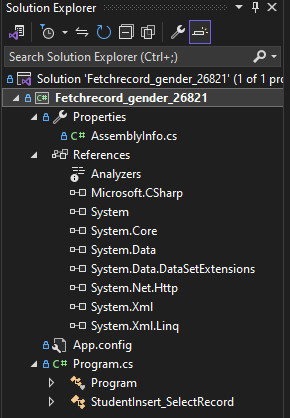
da.Fill(dt);

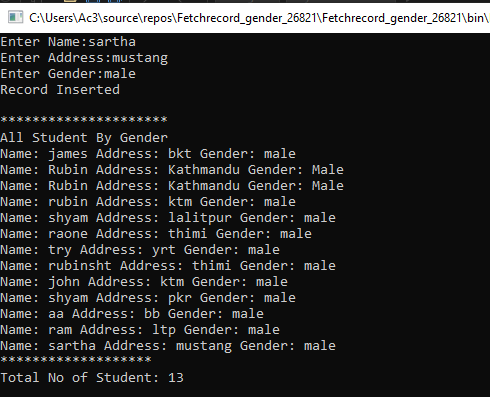
return dt;

}

}

}





**LAB 20**

Write a C# program to perform (CRUD) Operation from given table (tblStudent) with fields

(int id, nvarchar(50) name, int age, nvarchar(50) gender).

using System;

using System.Data;

using System.Data.SqlClient;

namespace CURD\_26821

{

internal class Program

{

static void Main(string[] args)

{

Student st = new Student();

Console.WriteLine("Enter Option");

Console.WriteLine("1 For Insert Student");

Console.WriteLine("2 For Update Student");

Console.WriteLine("3 For Delete Student");

Console.WriteLine("4 For Fetch All Student");

string option = Console.ReadLine();

switch (option)

{

case "1":

Console.Write("Enter Name: ");

string name = Console.ReadLine();

Console.Write("Enter Address: ");

string address = Console.ReadLine();

Console.Write("Enter Gender: ");

string gender = Console.ReadLine();

st.InsertStudent(name, address, gender);

Console.WriteLine("Record Inserted");

break;

case "2":

Console.Write("Enter Id To Update: ");

int id = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter Name: ");

string uname = Console.ReadLine();

Console.Write("Enter Address: ");

string uaddress = Console.ReadLine();

Console.Write("Enter Gender: ");

string ugender = Console.ReadLine();

st.UpdateStudent(uname, uaddress, ugender, id);

Console.WriteLine("Record Updated");

break;

case "3":

Console.Write("Enter Id To Delete: ");

int did = Convert.ToInt32(Console.ReadLine());

st.DeleteStudent(did);

Console.WriteLine("Record Deleted");

break;

case "4":

DataTable dt = st.DisplayStudentData();

for (int i = 0; i < dt.Rows.Count; i++)

{

Console.WriteLine("Name: {0} Address: {1} Gender: {2}",

dt.Rows[i]["Name"], dt.Rows[i]["Address"], dt.Rows[i]["Gender"]);

}

break;

default:

Console.WriteLine("Invalid Option");

break;

}

Console.ReadLine();

}

}

public class Student

{

private string connectionString = "Data Source=(localdb)\\MSSqlLocalDB;Database=SamriddhiDB;Integrated Security=true";

public void InsertStudent(string name, string address, string gender)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

string sql = "INSERT INTO tblStudent (Name, Address, Gender) VALUES (@name, @address, @gender)";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@name", name);

cmd.Parameters.AddWithValue("@address", address);

cmd.Parameters.AddWithValue("@gender", gender);

con.Open();

cmd.ExecuteNonQuery();

}

}

public void UpdateStudent(string name, string address, string gender, int id)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

string sql = "UPDATE tblStudent SET Name = @name, Address = @address, Gender = @gender WHERE Id = @id";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@name", name);

cmd.Parameters.AddWithValue("@address", address);

cmd.Parameters.AddWithValue("@gender", gender);

cmd.Parameters.AddWithValue("@id", id);

con.Open();

cmd.ExecuteNonQuery();

}

}

public void DeleteStudent(int id)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

string sql = "DELETE FROM tblStudent WHERE Id = @id";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@id", id);

con.Open();

cmd.ExecuteNonQuery();

}

}

public DataTable DisplayStudentData()

{

using (SqlConnection con = new SqlConnection(connectionString))

{

string sql = "SELECT \* FROM tblStudent";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

return dt;

}

}

}

}

