using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace StructEnumAbtsractClassInterfaceDemo

{

enum Choice { Addition , Subtarction , Product, Quotient, Remainder};

class Operations

{

static void Main()

{

int n1, n2, ch;

Console.WriteLine("Enter No1");

n1 = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter No2");

n2 = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter choice");

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

switch(ch)

{

case 1: Console.WriteLine(n1+n2); break;

case 2: Console.WriteLine(n1 - n2); break;

case 3: Console.WriteLine(n1 \* n2); break;

case 4: Console.WriteLine(n1 / n2); break;

case 5: Console.WriteLine(n1 % n2); break;

default: Console.WriteLine("Invalid Choice"); break;

}

}

}

}

With Enum

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace StructEnumAbtsractClassInterfaceDemo

{

enum Choice { Addition=1 , Subtarction , Product, Quotient, Remainder};

class OperationsWithEnum

{

static void Main()

{

int n1, n2, ch;

Console.WriteLine("Enter No1");

n1 = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter No2");

n2 = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter choice");

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

switch(ch)

{

case (int)Choice.Addition : Console.WriteLine(n1+n2); break;

case (int)Choice.Subtarction : Console.WriteLine(n1 - n2); break;

case (int)Choice.Product: Console.WriteLine(n1 \* n2); break;

case (int)Choice.Quotient: Console.WriteLine(n1 / n2); break;

case (int)Choice.Remainder: Console.WriteLine(n1 % n2); break;

default: Console.WriteLine("Invalid Choice"); break;

}

}

}

}

------------------  
Structure

using System;

namespace StructEnumAbtsractClassInterfaceDemo

{

struct Record

{

int id;

string name;

string address;

//public Record()

//{

//}

public Record(int id, string name, string address)

{

this.id = id;

this.name = name;

this.address = address;

}

public void Get()

{

Console.WriteLine("Enter Id");

id = Convert.ToByte(Console.ReadLine());

Console.WriteLine("ENter Address");

address = Console.ReadLine();

Console.WriteLine("ENter Name");

name = Console.ReadLine();

}

public void Display()

{

Console.WriteLine($"ID is {id}\nName is{name}\nAddress is {address}");

}

}

//struct Rec : Record

//{

//}

class Program

{

static void Main(string[] args)

{

Record record = new Record();

record.Get();

record.Display();

Employee employee = new Employee();

employee.GetDetails();

}

}

}

Class with Structure

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace StructEnumAbtsractClassInterfaceDemo

{

struct Address

{

string streetNo;

string city;

string state;

string pinCode;

public void GetAddressDetails()

{

Console.WriteLine("ENter StreetNo");

streetNo = Console.ReadLine();

Console.WriteLine("Enter City");

city = Console.ReadLine();

Console.WriteLine("Enter State");

state = Console.ReadLine();

}

}

struct DateOfBirth

{

public int day, month, year;

}

class Employee

{

int id;

string name;

Address address;

DateOfBirth dob;

public void GetDetails()

{

Console.WriteLine("Enter Id");

id = Convert.ToByte(Console.ReadLine());

Console.WriteLine("ENter NAme");

name = Console.ReadLine();

address.GetAddressDetails();

Console.WriteLine("Enter Date of Birth");

dob.day = byte.Parse(Console.ReadLine());

dob.month = byte.Parse(Console.ReadLine());

dob.year = byte.Parse(Console.ReadLine());

}

}

}

Without Abstract Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace StructEnumAbtsractClassInterfaceDemo

{ class Square

{

int side, area;

public void GetDimensions()

{

Console.WriteLine("Enter Value of side");

side = Byte.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = side \* side;

}

public void DisplayArea()

{

Console.WriteLine($"Area is {area}");

}

}

class Recangle

{

int l, w, area;

public void GetDimensions()

{

Console.WriteLine("Enter Value of length");

l = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter Value of width");

w = Byte.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = l \* w;

}

public void DisplayArea()

{

Console.WriteLine($"Area is {area}");

}

}

class Triangle

{

int b,h , area;

public void GetDimensions()

{

Console.WriteLine("Enter Value of base");

b = Byte.Parse(Console.ReadLine());

Console.WriteLine("Enter Value of heigth");

h = Byte.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = (int).5 \* b \* h;

}

public void DisplayArea()

{

Console.WriteLine($"Area is {area}");

}

}

class AreaWithoutAC

{

}

}

With Abstract Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace StructEnumAbtsractClassInterface

{

abstract class Shape

{

protected int dim1, dim2, area;

public abstract void GetDimensions();

public abstract void CalculateArea();

public void DisplayArea()

{

Console.WriteLine($"Area is {area}");

}

}

class Square : Shape

{

public override void CalculateArea()

{

area = dim1 \* dim1;

}

public override void GetDimensions()

{

Console.WriteLine("Enter value for side");

dim1 = byte.Parse(Console.ReadLine());

}

}

class Rectangle : Shape

{

public override void CalculateArea()

{

area = dim1 \* dim2;

}

public override void GetDimensions()

{

Console.WriteLine("Enter l");

dim1 = byte.Parse(Console.ReadLine());

Console.WriteLine("ENter w");

dim2 = byte.Parse(Console.ReadLine());

}

}

class AreaWithAC

{

static void Main()

{

//Shape shape = new Shape();

Square square = new Square();

square.GetDimensions();

square.CalculateArea();

square.DisplayArea();

Rectangle rectangle = new Rectangle();

rectangle.GetDimensions();

rectangle.CalculateArea();

rectangle.DisplayArea();

}

}

}