Enum > Whenever we have some options , in that case to represent those options, we can use enumeration

Class student {}

Struct student{}

Enum Options {addition, subtraction , product , division}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EmployeeManagementSystem

{

//class Customer

//{

// int id;

// public string name;

// public Customer() { }

// public Customer(int id, string name)

// {

// this.id = id;

// this.name = name;

// }

// public Customer(Customer onj)

// {

// id = onj.id;

// name = onj.name;

// }

//}

class Program

{

enum Options { addition, subtraction , product , division };

static void Main(string[] args)

{

int num1, num2, choice;

Console.WriteLine("Enter No1");

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

Console.WriteLine("Enter No2");

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

Console.WriteLine("Enter Choice");

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

switch(choice)

{

case 1: Console.WriteLine(num1+ num2);

break;

case 2:

Console.WriteLine(num1 - num2);

break;

case 3: Console.WriteLine(num1 \* num2);

break;

case 4: Console.WriteLine(num1 / num2);

break;

default:

Console.WriteLine("Invalid choice");

break;

}

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EmployeeManagementSystem

{

//class Customer

//{

// int id;

// public string name;

// public Customer() { }

// public Customer(int id, string name)

// {

// this.id = id;

// this.name = name;

// }

// public Customer(Customer onj)

// {

// id = onj.id;

// name = onj.name;

// }

//}

class Program

{

enum Options { addition, subtraction , product , division };

static void Main(string[] args)

{

int num1, num2, choice;

Console.WriteLine("Enter No1");

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

Console.WriteLine("Enter No2");

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

Console.WriteLine("Enter Choice");

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

switch(choice)

{

case (int)Options.addition: Console.WriteLine(num1+ num2);

break;

case (int)Options.subtraction:

Console.WriteLine(num1 - num2);

break;

case (int)Options.product: Console.WriteLine(num1 \* num2);

break;

case (int)Options.division: Console.WriteLine(num1 / num2);

break;

default:

Console.WriteLine("Invalid choice");

break;

}

}

}

}

In C#, an enum (or enumeration type) is used to assign constant names to a group of numeric integer values. It makes constant values more readable, for example, WeekDays.Monday is more readable then number 0 when referring to the day in a week.

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EmployeeManagementSystem

{

//class Customer

//{

// int id;

// public string name;

// public Customer() { }

// public Customer(int id, string name)

// {

// this.id = id;

// this.name = name;

// }

// public Customer(Customer onj)

// {

// id = onj.id;

// name = onj.name;

// }

//}

class Program

{

enum Options { addition=1, subtraction , product , division , plus =6, minus, multiply , quotient};

static void Main(string[] args)

{

int num1, num2, choice;

Console.WriteLine("Enter No1");

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

Console.WriteLine("Enter No2");

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

Console.WriteLine("Enter Choice");

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

switch(choice)

{

case (int)Options.addition:

case (int)Options.plus:

Console.WriteLine(num1 + num2);

break;

case (int)Options.subtraction:

case (int)Options.minus:

Console.WriteLine(num1 - num2);

break;

case (int)Options.product:

case (int) Options.multiply:

Console.WriteLine(num1 \* num2);

break;

case (int)Options.division:

case (int)Options.quotient:

Console.WriteLine(num1 / num2);

break;

default:

Console.WriteLine("Invalid choice");

break;

}

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EmployeeManagementSystem

{

//class Customer

//{

// int id;

// public string name;

// public Customer() { }

// public Customer(int id, string name)

// {

// this.id = id;

// this.name = name;

// }

// public Customer(Customer onj)

// {

// id = onj.id;

// name = onj.name;

// }

//}

class Program

{

enum Options :Int16 { addition=1, subtraction=300 , product , division , plus =6, minus, multiply , quotient};

static void Main(string[] args)

{

int num1, num2, choice;

Console.WriteLine("Enter No1");

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

Console.WriteLine("Enter No2");

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

Console.WriteLine("Enter Choice");

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

switch(choice)

{

case (int)Options.addition:

case (int)Options.plus:

Console.WriteLine(num1 + num2);

break;

case (int)Options.subtraction:

case (int)Options.minus:

Console.WriteLine(num1 - num2);

break;

case (int)Options.product:

case (int) Options.multiply:

Console.WriteLine(num1 \* num2);

break;

case (int)Options.division:

case (int)Options.quotient:

Console.WriteLine(num1 / num2);

break;

default:

Console.WriteLine("Invalid choice");

break;

}

}

}

}

<https://www.tutorialsteacher.com/csharp/csharp-enum>

What is File Handling?

When we want to work with Files or Directories on your system. In that case , we use File Handling concepts

System.IO namespace is used

File & FileInfo

Directory & DirectoryInfo

File & Directory are static classes

FileInfo & DirectoyInfo contains instance methods

STATIC MEAN WE DONT NEED OBJECT

Console.Write

Convert.ToByte

using System;

using System.IO;

class Program

{

static void Main()

{

// File.Create(@"E:\demo1.txt");

//File.WriteAllText(@"E:\demo1.txt", "This is a text file");

//File.AppendAllText(@"E:\demo1.txt", "This is second line");

//File.Copy(@"E:\demo1.txt", @"E:\demo2.txt");

//File.Copy(@"E:\demo1.txt", @"E:\demo2.txt" ,false);

////File.Delete(@"E:\demo2.txt");

//FileInfo fileInfo = new FileInfo(@"E:\demo3.txt");

//fileInfo.AppendText();

//fileInfo.CopyTo(@"E:\demo4.txt");

Console.WriteLine(File.ReadAllText(@"E:\demo1.txt"));

Directory.CreateDirectory(@"E:\DIR1DEmO");

string[] dirs = Directory.GetFiles(@"E:\trainings\cts");

foreach(string temp in dirs)

Console.WriteLine(temp);

}

}

DirectoryInfo info = new DirectoryInfo(@"E:\trainings\cts");

using System;

using System.IO;

class Program

{

static void Main()

{

FileStream fs = new FileStream(@"E:\tempDemo.txt", FileMode.Create, FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

Console.WriteLine("Enter some text");

sw.Write("Hello");

sw.Close();

fs.Close();

}

}

using System;

using System.IO;

class Program

{

static void Main()

{

FileStream fs = new FileStream(@"E:\tempDemo.txt", FileMode.Append, FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

string text = string.Empty;

string ch = "y";

while (ch == "y")

{

Console.WriteLine("Enter some text");

text = Console.ReadLine();

sw.WriteLine(text);

Console.WriteLine("Do you want to write some other text");

ch = Console.ReadLine();

}

sw.Close();

fs.Close();

fs = new FileStream(@"E:\tempdemo.txt", FileMode.Open, FileAccess.Read);

StreamReader sr = new StreamReader(fs);

string str = sr.ReadLine();

while(str != null)

{

Console.WriteLine(str);

str=sr.ReadLine();

}

sr.Close();

fs.Close();

}

}

Static Class : Is a class which contains all the members which are static

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace EmployeeManagementSystem

{

static class Customer

{

public static string Product;

static Customer()

{

Product = "P11";

}

static void Get() { }

static void Display() { }

}

}

OVRRIDDEN IS POSSIBLE IN INHERITANCE

PROTECTED IN INHERITANCE