C#

try parse - guarda um endereço de memoria mesmo que não tenhas valor para a variável.

Convert - Converte um tipo de dados base em outro tipo de dados base.

Array – guarda variáveis do mesmo tipo. int NomeDoArray{x,x,x,x,x} se der para colocar os valores no momento de declaração ,Int[] NomeDoArray = new int[x], declaração do array se não se souber os valores.

Strings – datatype que pode considerar como um pedaço de texto, o sinal “+” serve para juntar varias palavras ou variáveis na mesma string, concatena

List – É uma classe genérica definida no namespace System.Collections.Generic. Semelhante ao array, mas usada quando é necessária mais flexibilidade.

* Declarar Lista:

List<int> NomeDaLista = new List <int>();

Para adicionar valores a lista usa se o método Add() :

* NomeDaLista.Add(x);

Os métodos como as propriedades são colocadas á frente do ponto final, para utilizar o método colocamos parêntesis ()

Exemp. Asdfg.Add();

Metodo length

Array: Indica o numero de items

String: Indica o numero de caracteres

Exemp. Int[] array = new int[5]; ConsoleWriteLine(“Length = {0}, array.Length);

* Count (length mas para listas)

Lista: Desenvolve o numero

X – numero/valor

Exemp.

NomeDaLista.Add(“x”);

List<int> NomeDaLista = new List<int>();

Console.WriteLine(NomeDaLista.Count);

* Sort()

Array : Recebe o array e ordena

Array.Sort(NomeDoArray);

* IndexOf()

Int posiçãox = Array.IndexOf(NomeDoArray, 0);

* Contains() para listas , pesquisar dentro das listas

Exemp.

Console.WriteLine(NomeDaLista, Contains(x)) devolve true ou false

* Insert(), insere na lista

Exemp. NomeDaLista.Insert(3,6);

* Remove () , remove das listas

Exemp.

NomeDaLista.Remove(x) a primeira ocurrencia de x, onde aparece primeiro ou NomeDaLista.RemoveAll(x) devolve o numero de elementos removidos da lista

* Clear()
* Substring()

Recebe dois argumentos, o primeiro como índex da string a remover e o segundo como a quantidade de caracteres a remover

string message = “Hello World”;

string newMessage = message.Substring(2,5); remove e apresenta os caracteres presentes entre a posição dois e a posição 5 exemp. “llo “

* Split()

Metodo que divide a string baseado num separador q o utilizador define

string nomes = “x,y,w,z”;

string[] separador = {“,”};

string[] arrayNomes = nomes.split(separador,StringSplitOptions.None); (none exemplo caso não haja opções nenhumas)

.removeEmptyLines -> tira os espaços

Exemp.

string aaaa = “w,x,y,z”;

char[] separador = {“,”};

string[] ListNomes = aaaaaa.Split(separador, StringSplitOptions.EmptyEntries);

* foreach(string xxxxx in yyyyyy) (ciclo for percorrer)

Exemplos----------------------------------------------------------------------------------------------

Utilizar .Parse

string numero = "6.99"

int numeroConvertido = int.Parse(numero);

Console.WriteLine(numeroConvertido);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Utilizar .TryParse

string numero2 = "6,66";

int resultadoConvertido = 0;

int.TryParse(numero2, out resultadoConvertido);

Console.WriteLine(resultadoConvertido);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Utiliazr Convert.

string numero3 = 0

int resultadoConvertido2 = 0;

Convert.ToInt32(numero3);

Console.WriteLine(resultadoConvertido2);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//Extras//

int numeroDecimal = 999;

string teste = numero2.ToString;

Converter float para int

float aaa = 999.999;

int intA = Convert.ToInt32(aaa);

Console.WriteLine("inteiro convertido = "+intA);

Exemplos----------------------------------------------------------------------------------------------

Operadores de comparação

|| - ou – (x>y || y<x)

&&- e – (x>y && y<x)

Instruções de Controlo de fluxo

* if

If(condição){

}else if(condição 2){

}else{

}

* switch

switch(expression)

switch(expression)

{

case x:

case y:

default:

}

Ciclos(loops)

* for

For(int i = 0, i < 5,i++)

{

}

* foreach

Foreach(type variableName in arrayName)

{

}

* while

While(condition){

}

* do

Do{

}

While(condition);

Instruções de salto

* Break

Switch(x)

{

Qualquer coisa;

Default:

Qualquer coisa;

**Break**;

}

* Continue – salta um paço pa frente

For(int i = 0, i < 5,i++)

{

Console.WriteLine(“a”);

Continue;

Console.WriteLine(“b”);

}

Dá print ao b apenas

* Return - Devolve

Console.WriteLine("Indique o numero para calcular o cubo");

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

double resultado = CalculaCubo(numero);

Console.WriteLine(resultado);

static double CalculaCubo(int numeroCalcular)

{

**return** Math.Pow(numeroCalcular, 3);

}

* Throw

If (args.Lenght > 1)

{

Throw new ArgumentException(“Tem um argumento a mais”);

}

Gestão de erros e exceções

* Try…catch…

Try

{

Código;

}

Catch(erro)

{

Exceção

}

Finally - para introduzir frase no final do programa corrido corretamente ou nao

{

Exemp. Console.WriteLine("O Programa correu até ao fim");

}

* Classes

Keyword class

class pessoa{

//propriedades campos e métodos

}

* Campos

public- publica

private-apenas na class

protected – acessível apenas a classes e subclasses

internal- acessível apenas dentro da mesma compilação

class Pessoa{

private string nomeDaPessoa;

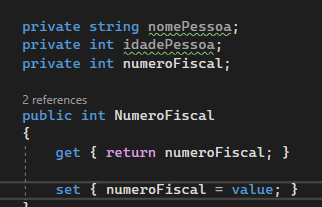
private int numeroFiscal;

}

* Propriedades

Propriedade para realizar uma função

public int NumeroFiscal janela Class

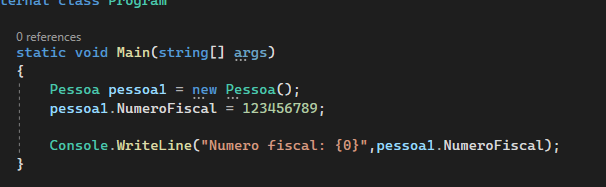
{

//get

//set

}

Getter

 Retorna o valor do campo

Janela principal

* Metodos = funções

Public double CalculaCubo(double valor)

{

Return Math.pow(valor,3);

}

* Overloading

Utilizaçao do mesmo método com o mesmo nome, mas com assinaturas diferentes

Public void BoasVindas(string nome)

{

Console.WriteLine(“Bem vindo {0}”,nome);

}

* Construtores

Public Pessoa(int nif)

{

numeroFiscal = nif;

Console.WriteLine($\*NIF

}

* Instanciar objetos

Devemos instanciar as classes no programa para a utilização dos objetos

NomeClasse nomeObjetoX = new NomeClass(argumentos);