Ejercicio - Métodos Anónimos

class Operaciones

{

public delegate void DelNumeros();

public DelNumeros numeros;

private int[] num = new int[10];

public int[] Num {get { return num; } set { num = value; } }

}

class Program

{

static void Main(string[] args)

{

Operaciones op = new Operaciones();

Random rnd = new Random();

op.numeros += delegate ()

{

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

{

op.Num[i] = rnd.Next(0, 100);

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine($"La posición {i} del arreglo es: {op.Num[i]}");

}

};

op.numeros += delegate ()

{

double prom = 0;

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

{

prom += op.Num[i];

}

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine($"\nEl promedio del arreglo es: {prom / op.Num.Length}");

};

op.numeros();

Console.ReadLine();

}

}

Ejercicio 2 – Eventos

namespace Ejercicio\_2\_\_\_Eventos

{

/// <summary>

/// Lógica de interacción para MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void btnAbrirVentana\_Click(object sender, RoutedEventArgs e)

{

Ventana2 ventana2 = new Ventana2();

ventana2.controlador += new Ventana2.Controlador(intercambiarVentana);

ventana2.Show();

}

public void intercambiarVentana(string cont)

{

ventana1.Title = cont;

}

}

}

namespace Ejercicio\_2\_\_\_Eventos

{

/// <summary>

/// Lógica de interacción para Ventana2.xaml

/// </summary>

public partial class Ventana2 : Window

{

public delegate void Controlador(string cont);

public event Controlador controlador;

public Ventana2()

{

InitializeComponent();

}

private void btnEnviar\_Click(object sender, RoutedEventArgs e)

{

controlador(txtTexto.Text);

}

public void intercambiarVentana(string vent)

{

if (controlador != null)

{

controlador(vent);

}

}

}

}

Ejercicio1 – Lambda

namespace Ejercicio1Lambda

{

public delegate bool Comparacion(int nota);

public class Estudiante

{

private string nombre;

private int nota;

public string Nombre { get; set; }

public int Nota { get; set; }

}

public class Curso

{

private Estudiante[] estudiantes = new Estudiante[5];

public void Cargar(int pos, Estudiante estudiante)

{

estudiantes[pos] = estudiante;

}

public void MostrarEstudiantes()

{

foreach (var estudiante in estudiantes)

{

Console.WriteLine($"Nombre: {estudiante.Nombre} - Nota: {estudiante.Nota}");

}

}

public void MostrarIf(Comparacion comparacion)

{

foreach (var estudiante in estudiantes)

{

if (comparacion(estudiante.Nota))

{

Console.WriteLine($"Nombre: {estudiante.Nombre} - Nota: {estudiante.Nota}");

}

}

}

}

class Program

{

static void Main(string[] args)

{

Curso c1 = new Curso();

for (int i = 0; i < 5; i++)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine($"Ingrese el nombre del estudiante #{(i+1)}: ");

string nameS = Console.ReadLine();

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine($"Ingrese la nota del estudiante #{ (i + 1)}: ");

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

c1.Cargar(i, new Estudiante

{

Nombre = nameS,

Nota = notaS

});

}

Console.ForegroundColor = ConsoleColor.Blue;

Console.WriteLine("Lista completa de estudiantes.");

c1.MostrarEstudiantes();

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine("Lista completa de estudiantes con nota mayor o igual a 51.");

c1.MostrarIf((nota) => nota >= 51);

Console.ForegroundColor = ConsoleColor.Magenta;

Console.WriteLine("Lista completa de estudiantes que tienen de nota 50.");

c1.MostrarIf((nota) => nota == 50);

Console.ReadKey();

}

}

}

Ejercicio 2 – Lambda

namespace Ejercicio2\_Lambda

{

delegate bool CadenaDelegado(char caracter);

class Program

{

public static string Filtrar(string cadena, CadenaDelegado del)

{

string fraseCadena = "";

foreach (var caracter in cadena)

if (del(caracter))

fraseCadena += caracter;

return fraseCadena.ToString();

}

static void Main(string[] args)

{

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("Ingrese una frase: ");

var cadena = Console.ReadLine();

Console.ForegroundColor = ConsoleColor.Magenta;

Console.WriteLine("Cadena: " + cadena);

var vocales = Program.Filtrar(cadena, caracter => caracter == 'a' || caracter == 'e' || caracter == 'i' || caracter == 'o' || caracter == 'u' ||

caracter == 'A' || caracter == 'E' || caracter == 'I' || caracter == 'O' || caracter == 'U');

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine($"La frase tiene como vocales: {vocales}");

var dig = Program.Filtrar(cadena, caracter => char.IsDigit(caracter));

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine($"La frase tiene como dígitos: {dig}");

var caracMin = Program.Filtrar(cadena, caracter => char.IsLower(caracter));

Console.ForegroundColor = ConsoleColor.Cyan;

Console.WriteLine($"La frase tiene como carácteres minúsculas: {caracMin}");

var caracNum = Program.Filtrar(cadena, caracter => char.IsLetter(caracter) || char.IsNumber(caracter));

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine("La frase tiene como carácteres y números: " + caracNum);

Console.ReadLine();

}

}

}