

# UNIVERSIDAD NACIONAL DE SAN AGUSTÍN

FACULTAD DE PRODUCCIÓN Y SERVICIOS  
ESCUELA PROFESIONAL DE INGENIERÍA DE  
SISTEMAS



Introducción a la Computación

Grupo: A

Profesor: Richart Smith Escobedo Quispe

Integrantes:

- Frank's Javier Vilca Quispe (100%)
- Gabriel Antony Mamani Cañari (100%)
- Hugo Villanueva Salamanca (100%)
- Carlo Rodrigo Diaz Portilla (100%)
- Alexander Angelo Quispe Torres (0%)

"Sea Google, Apple o software libre, tenemos competidores fantásticos y eso nos mantiene los pies sobre el suelo". - Bill Gates, Director ejecutivo de Microsoft Corporation

## **Presentación**

El ser humano para comunicarse el uno al otro necesita de un lenguaje que le permita transmitir y recibir información. Y la informática no queda excluida del uso del lenguaje, ya que esta es la manera de especificar las acciones que se desea que sean realizadas en la computadora.

Este es el idioma que usamos para controlar el comportamiento de una máquina, particularmente una computadora, y consiste en un conjunto de símbolos y reglas que definen su estructura y el significado de elementos y expresiones.

Actualmente una de las áreas más importantes en la industria y en el ámbito académico es el de la programación y dentro de este está el de la orientación a objetos que promete mejoras de amplio alcance en la forma de diseño, desarrollo y mantenimiento del software.

## **Resumen**

C# Es un lenguaje de programación multiparadigma desarrollado por la conocida compañía Microsoft. Fue desarrollado por Microsoft y forma parte de la familia C/C++. C# es considerado como una evolución y necesidad de ciertas circunstancias. Evolución por sus lenguajes antecesores que son el C y el C++ y necesidad a la hora en que la compañía tuvo problemas con la empresa creadora del lenguaje Java. Es por lo anterior que C Sharp (C#) presenta los atributos positivos de C++, Java y Visual Basic y los mejora otorgando un lenguaje fuerte y actualizado para los tiempos actuales.

C# fue diseñada para su uso en .NET, esta es una plataforma creada por Microsoft, la cual tiene como objetivo que los usuarios logren crear aplicaciones con sencillez, es decir, C# es un lenguaje de programación creado para diseñar aplicaciones en la plataforma .NET que, aunque no es el único lenguaje de programación que acepta .NET para realizar aplicaciones, C# si es el más recomendable y sencillo de usar.

Estas y más características se desarrollarán en este documento, de igual manera, se verán ejemplos del uso de este maravilloso lenguaje de programación que hoy en día está gustando a más desarrolladores.

Y para profundizar más en este lenguaje de programación, se desarrolló una aplicación que demuestra un poco la potencia de este lenguaje.

(Video explicativo de la app <https://youtu.be/BgcJwdvm6Pc> )

## **Expectativas**

- Comprender y valorar el lenguaje de programación C#. Desde lo más básico hasta lo más complejo, ampliando de esta manera nuestros conocimientos acerca de lenguajes de programación.
- Crear programas hasta los niveles más complejos en el programa C# y creando programas que faciliten al público.
- El poder conocer y utilizar las diferentes herramientas del lenguaje de programación c# para poder utilizarlo con destrezas y facilidad cada que lo necesitemos.

## **ÍNDICE**

Introducción al lenguaje .....	7
Historia .....	7
Paradigma .....	8
Aplicaciones .....	9
Ventajas .....	9
Desventajas .....	9
Ejemplos C# .....	10
Ejercicios C# .....	12
Aplicación C# .....	19
Recomendaciones .....	20
Conclusiones .....	21
Bibliografía .....	22

## INTRODUCCIÓN AL LENGUAJE:

### ➤ Historia:

Andres Hejlsber ingeniero de software, quien anteriormente había participado en el desarrollo de lenguajes como Turbo Pascal, Delphi y J + +. En 1999 decidió formar un equipo de trabajo para crear un nuevo lenguaje de programación, el cual hoy conocemos como c#. Sin embargo, en sus comienzos el nombre que se le dio fue Coll( C oriented language), lo que en español traduciremos como un lenguaje de programación orientado a objetos.

Pero hoy en día su nombre "C#" es un juego de palabras puesto que "C#" musicalmente significa "do sostenido" donde el signo # indica un tono más alto. Lo cual se puede decir que es una metáfora sobre su antecesor C++. También el símbolo # Puede hacer alusión a la unión de cuatro +, siendo el sentido de serie de C.

### ¿Qué es C#?

En inglés es pronunciado como "C Sharp, en español como "C Almohadilla" desarrollado por Microsoft con el objetivo de permitir a los desarrolladores crear una multitud de aplicaciones ejecutadas en .NET Framework , es la programación orientada a objetos, es una rama de la informática que usa como su propio nombre indica los objetos y las interacciones de estos para diseñar aplicaciones y programas informáticos. Cabe destacar que un objeto en programación es una entidad que combina el estado , comportamiento o método (las que define qué operaciones puede hacer el objeto) e identidad (es el factor diferenciador de los otros objetos).



## ➤ Paradigma:

Inicialmente c# tuvo un paradigma estrictamente orientado a objetos pero con el paso de sus actualizaciones tuvo seis paradigmas más.

- Estructurado: Es una técnica que se utiliza para escribir los programas de manera clara, para lo que se utilizan 3 técnicas.
  - ➔ Secuencia: Las instrucciones no se ejecutan hasta que la anterior haya finalizado.
  - ➔ Selección: Que se compruebe una instrucción o un conjunto según una condición y tomará uno de los caminos para ejecutarse.
  - ➔ Iteración: Que se repita una ejecución mientras se cumpla una condición.
- Imperativo: Los programas imperativos son un conjunto de instrucciones que le indican a la computadora cómo realizar una tarea.
- Programación orientada a objetos: Sus interacciones para el programa está basado en 3 distintas técnicas.
  - ➔ Herencia: Las clases se relacionan teniendo cierta jerarquía de clasificación. Los objetos heredan propiedades y el comportamiento de clases a la que pertenecen.
  - ➔ Abstracción: Los objetos en el sistema sirven como modelo de un agente abstracto puede realizar trabajo, informar y cambiar su estado, y comunicarse con otros objetos sin revelar cómo se implementan estas características.
  - ➔ Polimorfismo: Comportamientos diferentes, asociados a objetos distintos, pueden tener nombres similares y al usar ese nombre también se utilizará el comportamiento correspondiente.
  - ➔ Encapsulamiento: Reunir todos los elementos que pueden considerarse pertenecientes a una misma entidad con el mismo nivel de abstracción.
- Programación dirigida por eventos: La estructura como la ejecución de los programas va determinada por los sucesos que ocurran dentro del sistema, definidos por el usuario.
- Programación funcional: Se basa en la utilización de funciones aritméticas que no manejan datos mutables o de estado.
- Programación genérica: Está más centrada en algoritmos que en datos la idea pretende generalizar las funciones y pueden ser utilizadas en más de una ocasión.
- Reflexión: Es la capacidad que tiene un programa de ordenador para observar y opcionalmente modificar su estructura.



### ➤ **Aplicaciones:**

No existe un Top de lenguajes de programación en donde C# (C Sharp) no esté en los primeros lugares. En esta ocasión les presentare algunos de los proyectos más grandes en los que se utilizó C#.

-Skype para Windows Phone.\_ C# Es óptimo para el desarrollo de aplicaciones móviles en Windows Phone. Está optimizado para una muy buena persistencia a la hora de conectarse a la base de datos y darle funcionalidad a la aplicación.

-Microsoft Visual Studio. Esta es una herramienta muy potente para el desarrollo de software profesional.

-Unity .Es un motor de videojuego multiplataforma creado por Unity Technologies.

### ➤ **Ventajas:**

- Tipos de datos: Existe un rango más amplio y definido de tipos de datos que los que se encuentran en otros lenguajes como en c,c++ o Java.
- Pase de parámetros: Se puede declarar a los métodos para que acepten un número variable de parámetros de forma predeterminada.
- propiedades: Un objeto tiene intrínsecamente propiedades, debido a que las clases C# pueden ser utilizadas como objetos, C# permite la declaración de propiedades dentro de cualquier clase.
- control de versiones: Permite mantener múltiples versiones de clases en forma binaria, colocándolas en diferentes espacios de nombres. Esto permite utilizar versiones nuevas y anteriores de software.
- Lenguaje seguro: Es un lenguaje seguro, lo que implica varios elementos. Por ejemplo, usted no puede utilizar variables no inicializadas. En C++, es fácil declarar una variable y después revisar su valor; lo que sea que esté en la dirección de memoria dada a esa variable se mostraría, y esto podría causar problemas en una aplicación. En cambio, el compilador de C# le notificará si intenta utilizar una variable antes de inicializarla con algún valor válido.
- Buen manejo de memoria: Remueve problemas de manejo de memoria al desarrollador al usar el esquema de recolección de basura de .NET. Los ítems sin referencia son marcados para la recolección de basura y el marco de datos puede reclamar luego.

### ➤ **Desventajas:**

- Para tan solo la instalación se necesita alrededor de 4 gigas de espacio libre
- Para alguien que está comenzando en la programación se le hará muy complicado de aprender.

## Ejemplos de c#

<https://github.com/Gabicho258/TrabajoJC/tree/main/Ejemplos>

1) Hola mundo :

```
ejercicio01
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ejercicio01
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
            Console.ReadKey();
        }
    }
}
```

```
C:\Users\HP\source\repos\ejercicio01\ejercicio01
Hello World!
```

2) Variables:

```
using System;
namespace ejercicio2
{
    //variables y tipos de datos
    class Program
    {
        static void Main(string[] args)
        {
            string nombre = "Javier";
            int numero = 1;
            bool respuesta = true;
            double talla = 1.80;
            char b = default;
            char abecedario = b ;
            //La variable var es muy flexible por lo que puede
            //tomar distintos valores
            var nume = 1000; //entero
            var palabra = "cocina"; //string
            var peso = 10.8; //decimal
            var amor = true; //boolean
        }
    }
}
```

### 3) Estructura control(if, else):

```
1 using System;
2 namespace ConsoleApp1
3 {
4     O referencias
5     class Program
6     {
7         O referencias
8         static void Main(string[] args)
9         {
10             //Aqui le daremos los valores a los numeros
11             int num1 = 15, num2 = 50;
12             //veremos que numero es mayor para que se cumpla la condicion
13             if (num1 >= num2)
14             {
15                 Console.WriteLine("El primer numero es el mayor");
16             }
17             else
18             {
19                 Console.WriteLine("El segundo numero es el mayor");
20             }
21             Console.ReadKey();
22         }
23     }
24 }
```

C:\Users\HP\source\repos\ConsoleApp1\ConsoleApp

El segundo numero es el mayor

### 4) Estructura repetitiva while():

```
1 using System;
2 namespace ConsoleApp1
3 {
4     O referencias
5     class Program
6     {
7         O referencias
8         static void Main(string[] args)
9         {
10             //El programa escribira todos los numeros menores a 100
11             int x = 1;
12             while (x <= 100)
13             {
14                 Console.Write(x);
15                 Console.Write(" ; ");
16                 x++;
17                 //cada vez que escriba un numero se sumara mas uno hasta que llegue a 100
18             }
19             Console.ReadKey();
20         }
21     }
22 }
```

C:\Users\HP\source\repos\ConsoleApp1\ConsoleApp1\bin\Debug\netcoreapp3.1\ConsoleApp1.exe

1 ; 2 ; 3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9 ; 10 ; 11 ; 12 ; 13 ; 14 ; 15 ; 16 ; 17 ; 18 ; 19 ; 20 ; 21 ; 22 ; 23 ; 24 ; 25 ; 26 ;  
27 ; 28 ; 29 ; 30 ; 31 ; 32 ; 33 ; 34 ; 35 ; 36 ; 37 ; 38 ; 39 ; 40 ; 41 ; 42 ; 43 ; 44 ; 45 ; 46 ; 47 ; 48 ; 49 ; 50 ;  
51 ; 52 ; 53 ; 54 ; 55 ; 56 ; 57 ; 58 ; 59 ; 60 ; 61 ; 62 ; 63 ; 64 ; 65 ; 66 ; 67 ; 68 ; 69 ; 70 ; 71 ; 72 ; 73 ; 74 ;  
75 ; 76 ; 77 ; 78 ; 79 ; 80 ; 81 ; 82 ; 83 ; 84 ; 85 ; 86 ; 87 ; 88 ; 89 ; 90 ; 91 ; 92 ; 93 ; 94 ; 95 ; 96 ; 97 ; 98 ;  
99 ; 100 ;

## Ejercicios:

<https://github.com/Gabicho258/TrabajoIC/tree/main/Ejercicios>

### Ejercicios básicos:

1. Programa que permite determinar los divisores de un número entero.

```
1 //Autor: Carlo Rodrigo Diaz Portilla
2 using System;
3
4 namespace Basico1
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             int num;
11             Console.Write("Ingrese el número a evaluar: ");
12             num = int.Parse(Console.ReadLine());
13             for (int i = 1; i <= (num / 2); i++)
14             {
15                 if (num % i == 0)
16                     Console.Write(i + ", ");
17             }
18             Console.Write(num);
19         }
20     }
21 }
```

Ejecución:

Consola de depuración de Microsoft Visual Studio

```
Ingrese el número a evaluar: 40
1, 2, 4, 5, 8, 10, 20, 40
```

2. Programa que permite saber si el número entero ingresado es primo.

```
1 //Autor: Carlo Rodrigo Diaz Portilla
2 using System;
3
4 namespace Basico2
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             int num;
11             bool primo = true;
12             Console.Write("¿Número primo?: ");
13             num = int.Parse(Console.ReadLine());
14             for (int i = 2; i <= (num / 2); i++)
15             {
16                 if (num % i == 0)
17                 {
18                     primo = false;
19                     break;
20                 }
21             }
22             if (primo)
23             {
24                 Console.WriteLine(num + " es un número primo.");
25             }
26             else
27             {
28                 Console.WriteLine(num + " no es un número primo.");
29             }
30         }
31     }
32 }
```

Ejecución:

C:\> Consola de depuración de Microsoft Visual Studio

```
¿Número primo?: 100
100 no es un número primo.
```

C:\> Consola de depuración de Microsoft Visual Studio

```
¿Número primo?: 101
101 es un número primo.
```

## Ejercicios medios:

1. Programa que calcula el Fibonacci de un número. Muestra los números anteriores.

```
1 //Autor: Carlo Rodrigo Diaz Portilla
2 using System;
3
4 namespace Mediol
5 {
6     Oreferencias
7     class Program
8     {
9         Oreferencias
10        static void Main(string[] args)
11        {
12            int num,numFibonacci;
13            Console.Write("Ingrese el número para obtener Fibonacci: ");
14            num = int.Parse(Console.ReadLine());
15            numFibonacci = fibonacci(num);
16            Console.WriteLine("El número Fibonacci de " + num + " es: " + numFibonacci);
17        }
18        Oreferencias
19        static int fibonacci(int num)
20        {
21            if (num == 1 || num == 0)
22                return num;
23            else
24                return fibonacci(num-1)+fibonacci(num-2);
25        }
26    }
27 }
```

Ejecución:

 Consola de depuración de Microsoft Visual Studio

```
Ingrese el número para obtener Fibonacci: 10
El número Fibonacci de 10 es: 55
```

2. Simulación de juego de dados. El jugador lanza dos dados y la computadora otros dos, gana el que tenga la mayor cantidad.

```
1 //Autor: Carlo Rodrigo Diaz Portilla.
2 using System;
3
4 namespace Medio2
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             int dadoJugador, dadoComputadora, respuesta;
11             for (; ; )
12             {
13                 dadoJugador = lanzarDados();
14                 dadoComputadora = lanzarDados();
15                 if (dadoJugador > dadoComputadora)
16                     Console.WriteLine("Has ganado!");
17                 else
18                     Console.WriteLine("Has perdido!");
19                 Console.WriteLine("Resultados: \nJugador: {0}\nComputadora: {1}", dadoJugador, dadoComputadora);
20                 Console.WriteLine("¿Desea volver a jugar?<1>Si <2>No");
21                 respuesta = int.Parse(Console.ReadLine());
22                 if (respuesta == 1)
23                     continue;
24                 else if (respuesta == 2)
25                     break;
26             }
27         }
28         static int lanzarDados()
29         {
30             Random random = new Random();
31             return random.Next(12) + 1;
32         }
33     }
34 }
```

Ejecución:

Consola de depuración de Microsoft Visual Studio

```
Has perdido!
Resultados:
Jugador: 4
Computadora: 5
¿Desea volver a jugar?<1>Si <2>No
1
Has perdido!
Resultados:
Jugador: 3
Computadora: 3
¿Desea volver a jugar?<1>Si <2>No
1
Has ganado!
Resultados:
Jugador: 10
Computadora: 6
¿Desea volver a jugar?<1>Si <2>No
2
```

## Ejercicios Avanzados:

1. Suma, Resta, Multiplicación Punto de una matriz n x m.

```
1 //Autor: Carlo Rodrigo Diaz Portilla
2 using System;
3
4 namespace Avanzado1
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             int filas, columnas, accion;
11             Console.WriteLine("Ingresa las filas de la matriz: ");
12             filas = int.Parse(Console.ReadLine());
13             Console.WriteLine("Ingresa las columnas de la matriz: ");
14             columnas = int.Parse(Console.ReadLine());
15             int[,] _matriz1 = matriz(filas, columnas);
16             int[,] _matriz2 = matriz(filas, columnas);
17             Console.WriteLine("Matriz 1 generada aleatoriamente: ");
18             imprimirMatriz(_matriz1, filas, columnas);
19             Console.WriteLine("Matriz 2 generada aleatoriamente: ");
20             imprimirMatriz(_matriz2, filas, columnas);
21             for (; ; )
22             {
23                 Console.WriteLine("<1>Suma <2>Resta <3>Multiplicación Punto <4>Finalizar: ");
24                 accion = int.Parse(Console.ReadLine());
25                 if (accion == 4)
26                     break;
27                 switch (accion)
28                 {
29                     case 1: int[,] _matrizSuma = suma(_matriz1, _matriz2, filas, columnas);
30                             imprimirMatriz(_matrizSuma, filas, columnas);
31                             break;
32                     case 2: int[,] _matrizResta = resta(_matriz1, _matriz2, filas, columnas);
33                             imprimirMatriz(_matrizResta, filas, columnas);
34                             break;
35                     case 3:
36                         int[,] _matrizMultiplicacionPunto = multiplicacionPunto(_matriz1, _matriz2, filas, columnas);
37                         imprimirMatriz(_matrizMultiplicacionPunto, filas, columnas);
38                         break;
39                 }
40             }
41         }
42     }
43 }
```

```
42 static int[,] matriz(int filas, int columnas)
43 {
44     int[,] _matriz = new int[filas, columnas];
45     for (int i = 0; i < filas; i++)
46         for (int j = 0; j < columnas; j++)
47             _matriz[i, j] = completarMatriz();
48     return _matriz;
49 }
50 static int completarMatriz()
51 {
52     Random random = new Random();
53     return random.Next(30) + 1;
54 }
55 static int[,] suma(int[,] matriz1, int[,] matriz2, int filas, int columnas)
56 {
57     int[,] _matriz = new int[filas, columnas];
58     for (int i = 0; i < filas; i++)
59         for (int j = 0; j < columnas; j++)
60             _matriz[i, j] = matriz1[i, j] + matriz2[i, j];
61     return _matriz;
62 }
63 static int[,] resta(int[,] matriz1, int[,] matriz2, int filas, int columnas)
64 {
65     int[,] _matriz = new int[filas, columnas];
66     for (int i = 0; i < filas; i++)
67         for (int j = 0; j < columnas; j++)
68             _matriz[i, j] = matriz1[i, j] - matriz2[i, j];
69     return _matriz;
70 }
71 static int[,] multiplicacionPunto(int[,] matriz1, int[,] matriz2, int filas, int columnas)
72 {
73     int[,] _matriz = new int[filas, columnas];
74     for (int i = 0; i < filas; i++)
75         for (int j = 0; j < columnas; j++)
76             _matriz[i, j] = matriz1[i, j] * matriz2[i, j];
77     return _matriz;
78 }
```



```

79  static void imprimirMatriz(int [,] matriz,int filas,int columnas)
80  {
81      for (int i = 0; i < filas; i++)
82      {
83          for (int j = 0; j < columnas; j++)
84              Console.Write("(" + matriz[i, j] + " ");
85          Console.WriteLine("");
86      }
87  }
88  }
89  }

```

Ejecución:

```

CA Consola de depuración de Microsoft Visual Studio
Ingresa las filas de la matriz: 3
Ingresa las columnas de la matriz: 3
Matriz 1 generada aleatoriamente:
(30)(11)(9)
(13)(2)(22)
(2)(16)(21)
Matriz 2 generada aleatoriamente:
(20)(1)(16)
(21)(28)(29)
(18)(11)(11)
<1>Suma <2>Resta <3>MultiplicaciónPunto <4>Finalizar: 1
(50)(12)(25)
(34)(30)(51)
(20)(27)(32)
<1>Suma <2>Resta <3>MultiplicaciónPunto <4>Finalizar: 2
(10)(10)(-7)
(-8)(-26)(-7)
(-16)(5)(10)
<1>Suma <2>Resta <3>MultiplicaciónPunto <4>Finalizar: 3
(600)(11)(144)
(273)(56)(638)
(36)(176)(231)
<1>Suma <2>Resta <3>MultiplicaciónPunto <4>Finalizar: 4

```

2. Multiplicación de matrices  $m \times n$  y  $n \times p$  (Diferente a multiplicación punto de dos matrices).

```

1 //Autor: Carlo Rodrigo Diaz Portilla
2 using System;
3
4 namespace Avanzado2
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             int fila1, columnaFila2, columna2;
11             Console.Write("Ingrese filas de la primera matriz: ");
12             fila1 = int.Parse(Console.ReadLine());
13             Console.Write("Ingrese columnas de la primera matriz y filas de la segunda matriz: ");
14             columnaFila2 = int.Parse(Console.ReadLine());
15             Console.Write("Ingrese columnas de la segunda matriz: ");
16             columna2 = int.Parse(Console.ReadLine());
17             int[,] matriz1 = rellenarMatriz(fila1, columnaFila2);
18             int[,] matriz2 = rellenarMatriz(columnaFila2, columna2);
19             int[,] multiplicacion = matrizMultiplicacion(matriz1, matriz2, fila1, columnaFila2, columna2);
20             Console.WriteLine("Matriz 1: ");
21             imprimirMatriz(matriz1, fila1, columnaFila2);
22             Console.WriteLine("Matriz 2: ");
23             imprimirMatriz(matriz2, columnaFila2, columna2);
24             Console.WriteLine("Matriz multiplicación: ");
25             imprimirMatriz(multiplicacion, fila1, columna2);
26         }
27
28         static int[,] rellenarMatriz(int filas, int columnas)
29         {
30             Random random = new Random();
31             int[,] matriz = new int[filas, columnas];
32             for (int i = 0; i < filas; i++)
33                 for (int j = 0; j < columnas; j++)
34                     matriz[i, j] = random.Next(30) + 1;
35             return matriz;
36         }
37
38         static int[,] matrizMultiplicacion(int[,] matriz1, int[,] matriz2, int fila1, int columnaFila2, int columna2)
39         {
40             int[,] matriz = new int[fila1, columna2];
41             for (int k = 0; k < columna2; k++)
42                 for (int i = 0; i < columnaFila2; i++)
43                     for (int j = 0; j < columna2; j++)
44                         matriz[i, j] += matriz1[i, k] * matriz2[k, j];
45             return matriz;
46         }
47
48         static void imprimirMatriz(int[,] matriz, int filas, int columnas)
49         {
50             for (int i = 0; i < filas; i++)
51             {
52                 for (int j = 0; j < columnas; j++)
53                     Console.Write("(" + matriz[i, j] + " ");
54                 Console.WriteLine("");
55             }
56         }
57     }
58 }

```

Ejecución:

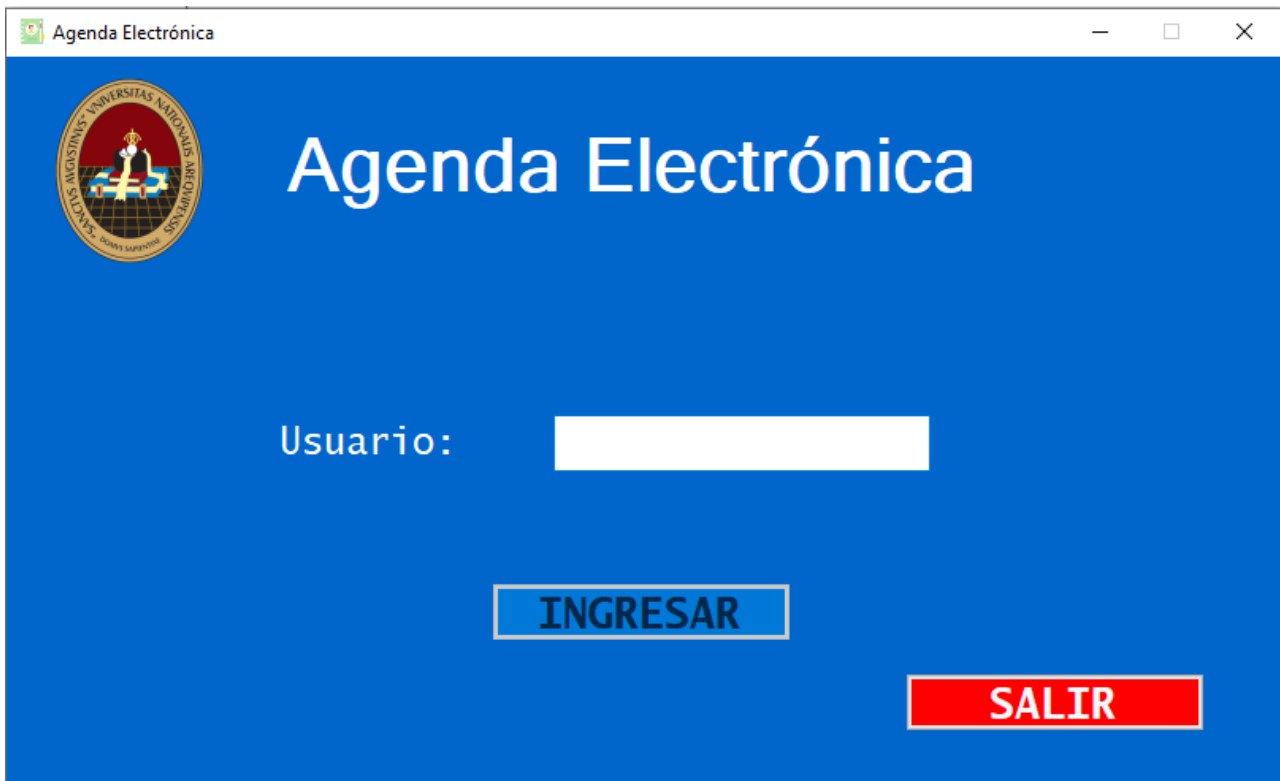
```

Consola de depuración de Microsoft Visual Studio
Ingrese filas de la primera matriz: 2
Ingrese columnas de la primera matriz y filas de la segunda matriz: 3
Ingrese columnas de la segunda matriz: 4
Matriz 1:
(28)(8)(16)
(12)(24)(30)
Matriz 2:
(2)(8)(6)(8)
(3)(3)(16)(8)
(24)(23)(1)(19)
Matriz multiplicación:
(464)(616)(312)(592)
(816)(858)(486)(858)


```

## Aplicación

<https://github.com/Gabicho258/TrabajoIC/tree/main/Aplicacion>



Agenda Electrónica



# Agenda Electrónica

Usuario:

**INGRESAR**

**SALIR**



Agenda Electrónica

## Bienvenid@, GrupoCsharp

**Añadir contacto**

**Eliminar contacto**

**Salir**

### Lista de contactos

Nombre: Gabriel	Teléfono: 999999999
Nombre: Frank's	Teléfono: 999999999
Nombre: Carlo	Teléfono: 999999999
Nombre: Hugo	Teléfono: 999999999

## **Recomendaciones:**

- Tener cuidado con la parte del input del programa, "Console.ReadLine()" lee el input como String, entonces para evitar este problema podemos hacer uso de las clases wrapper para ingresarlos como enteros o el tipo de dato que se desee(int.Parse())..
- Al momento de concatenar en C#, existe una manera que resulta más fácil, se trata de las llaves { }, podemos simplemente colocar estas llaves y dentro un número el cual se inicia desde el cero donde se van a ir concatenando los diversos tipos de datos.
- Al estar programando en C# después de tener algo de experiencia en Java te encontrarás con las diferencias de la clase String en Java como objetos por referencia, en C# es diferente ya que se "String" y "string" son diferentes, este último como dato primitivo entonces para declarar y crear se hace uso de "string" y no "String" como es el caso de Java.
- En el caso de C# al declarar y crear arreglos bidimensionales utilizaremos una coma en medio para definir que es bidimensional [filas,columnas] a diferencia de Java donde sería [filas] [columnas].
- Programar en C# puede ser sencillo si es que tenemos experiencia en Java, C++ o C, salvo algunas excepciones que son bastante fáciles de averiguar por cuenta propia.
- Ser paciente. Como todo lenguaje de programación, adaptarse y comprender la sintaxis del lenguaje llevará cierto tiempo en función del esfuerzo que se le ponga. Es por ello, que se requiere de paciencia para poder aprender y descubrir todo lo que nos otorga este potente lenguaje de programación.

## **Conclusiones:**

- C# Como lenguaje de programación que puede ser utilizado para la programación orientada a objetos, posee una sintaxis sencilla y fácil de comprender por lo que beneficia a usuarios con experiencia previa y a los que no poseen mucha experiencia.
- Como se pudo ver, el lenguaje de programación C# es un lenguaje útil para la ejecución y desarrollo dentro de múltiples ramas de la programación y especialmente desde en su el desarrollo de aplicaciones de escritorio y desarrollo de videojuegos con el motor Unity.
- Con C# Es más sencillo programar teniendo en cuenta que es necesario tener una base como de (java).
- C# Es uno de los programas más fáciles para desarrollar programas.
- El programa c# tiene lo mejor de los lenguajes de programación como vendría a ser C, C++, JAVA y Delphi.
- Las herramientas que te brinda el programa son de fácil comprensión si tienes experiencia con otros lenguajes, además que el programa es más funcional y moderno.

## **BIBLIOGRAFÍA**

- Es.wikipedia.org. 2021. *C Sharp - Wikipedia, la enciclopedia libre*. [online] Available at: <[https://es.wikipedia.org/wiki/C\\_Sharp](https://es.wikipedia.org/wiki/C_Sharp)> [Accessed 28 July 2021].
- Docs.microsoft.com. 2021. *C# docs - get started, tutorials, reference..* [online] Available at: <<https://docs.microsoft.com/en-us/dotnet/csharp/>> [Accessed 25 July 2021].
- W3schools.com. 2021. C# Tutorial (C Sharp). [online] Available at: <<https://www.w3schools.com/cs/index.php>> [Accessed 26 July 2021].
- Negociosyestrategia.com. 2021. C#. Qué es y para qué se utiliza | n+e. [online] Available at: <<https://negociosyestrategia.com/blog/que-es-csharp/>> [Accessed 24 July 2021].
- Tokio School. 2021. ¿Sabes qué es C#? ¡Conoce este lenguaje de programación!. [online] Available at: <<https://www.tokioschool.com/noticias/c-que-es/#:~:text=C%23%20tiene%20sus%20or%C3%ADgenes%20en,llegar%20a%20la%20versi%C3%B3n%20actual>> [Accessed 25 July 2021].
- nishiprog. 2021. lenguaje multiparadigma C#. [online] Available at: <<https://nishiprog.wordpress.com/2011/10/08/lenguaje-multiparadigma-c/>> [Accessed 24 July 2021].
- Instituto Americano. 2021. C# Paradigmas de programación. [online] Available at: <<https://www.institutoamericano.es/c-paradigmas-de-programacion/>> [Accessed 24 July 2021].
- INFORMÁTICA Y TECNOLOGÍA. 2021. que es c# | ventajas, y sus propiedades - códigos informáticos. [online] Available at: <<https://www.codigosinformaticos.com/curso-por-lenguaje/c/que-es-c/>> [Accessed 26 July 2021].
- Programacion1abundiz.blogspot.com. 2021. VENTAJAS DEL C# y DESVENTAJAS. [online] Available at: <<http://programacion1abundiz.blogspot.com/2009/09/ventajas-del-c-y-desventajas.html>> [Accessed 27 July 2021].