INSTITUTO KHIPU

Semestre III

Sesión 03

PROGRAMA DE ESTUDIOS

DESARROLLO DE SISTEMAS DE INFORMACIÓN

HERRAMIENTAS DE PROGRAMACION – C#

Tema

TIPOS DE METODOS: MODIFICADOR PRIVATE Y MÉTODOS QUE NO RETORNA ELEMENTOS.

TIPOS DE METODOS

Modificador prívate y métodos que no retorna elementos.

En **Programación Orientada a Objetos (POO),** los métodos con el modificador **private** son aquellos que solo pueden ser accedidos desde dentro de la propia clase. Esto significa que no pueden ser llamados desde fuera de la clase, ni siquiera por clases que heredan de ella. Se utilizan para encapsular la lógica interna de la clase y proteger datos o comportamientos que no deben ser accesibles directamente.

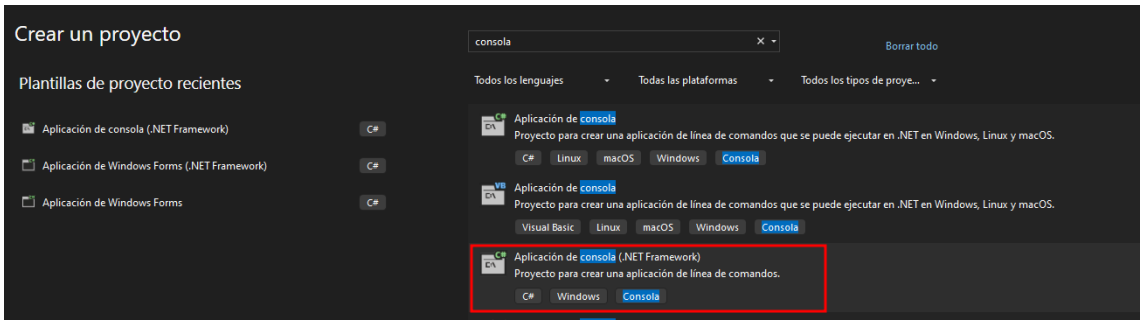
Los métodos que no retornan elementos son métodos de tipo **void** en lenguajes como C#. Estos métodos realizan una tarea, pero no devuelven ningún valor al ser llamados.

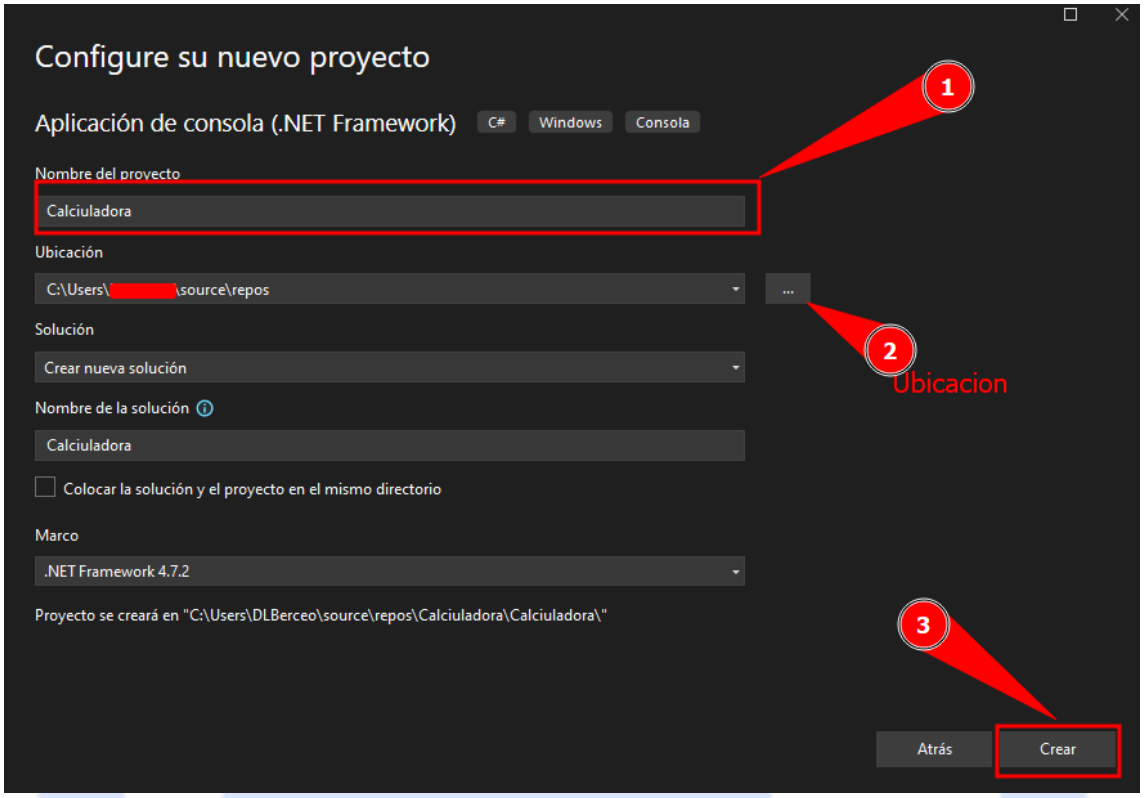
Modificador private (Métodos privados)

Los métodos privados son aquellos que solo pueden ser utilizados dentro de la misma clase. A menudo se utilizan para tareas auxiliares o para encapsular detalles internos que no deberían ser accesibles desde fuera de la clase.

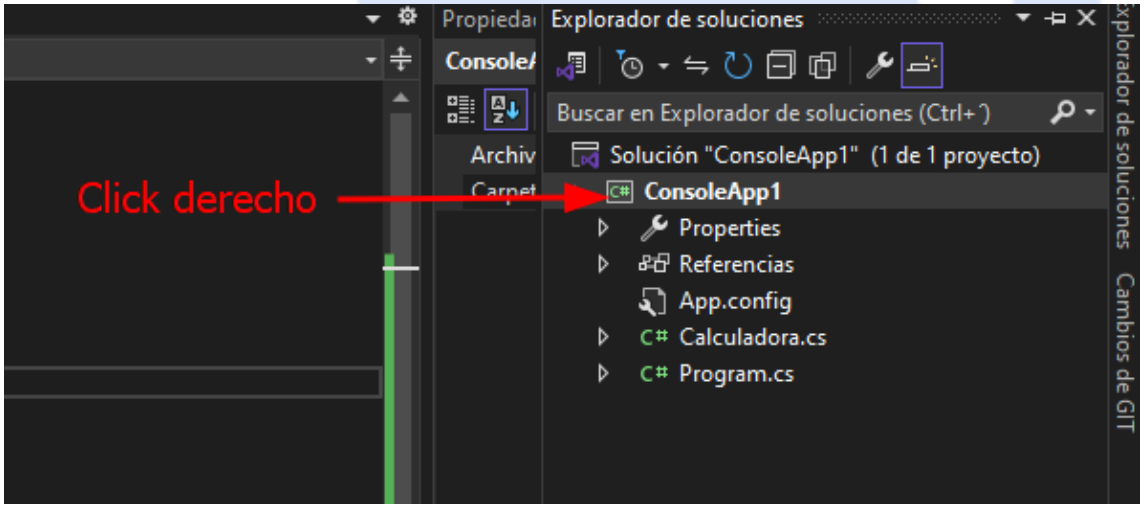
EJEMPLO PRACTICO

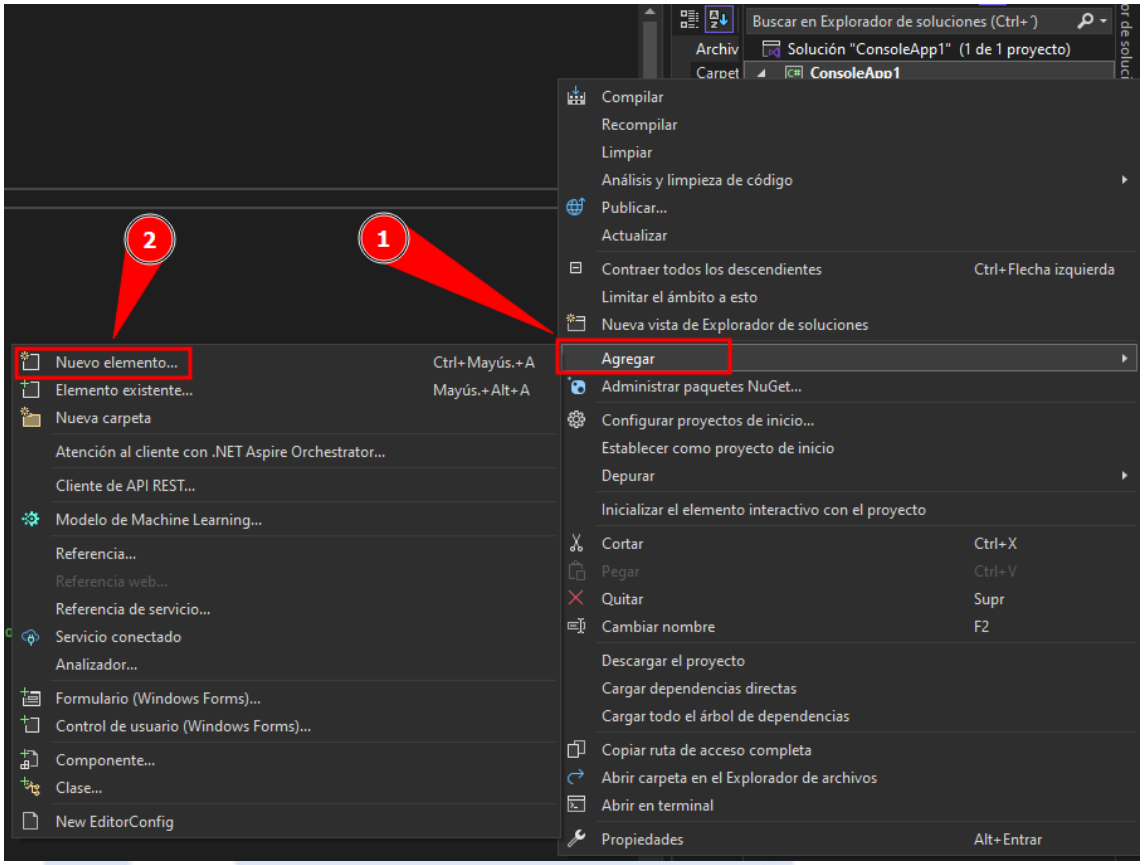
Paso 1.- creamos un nuevo proyecto en visual studio – aplicación de consola

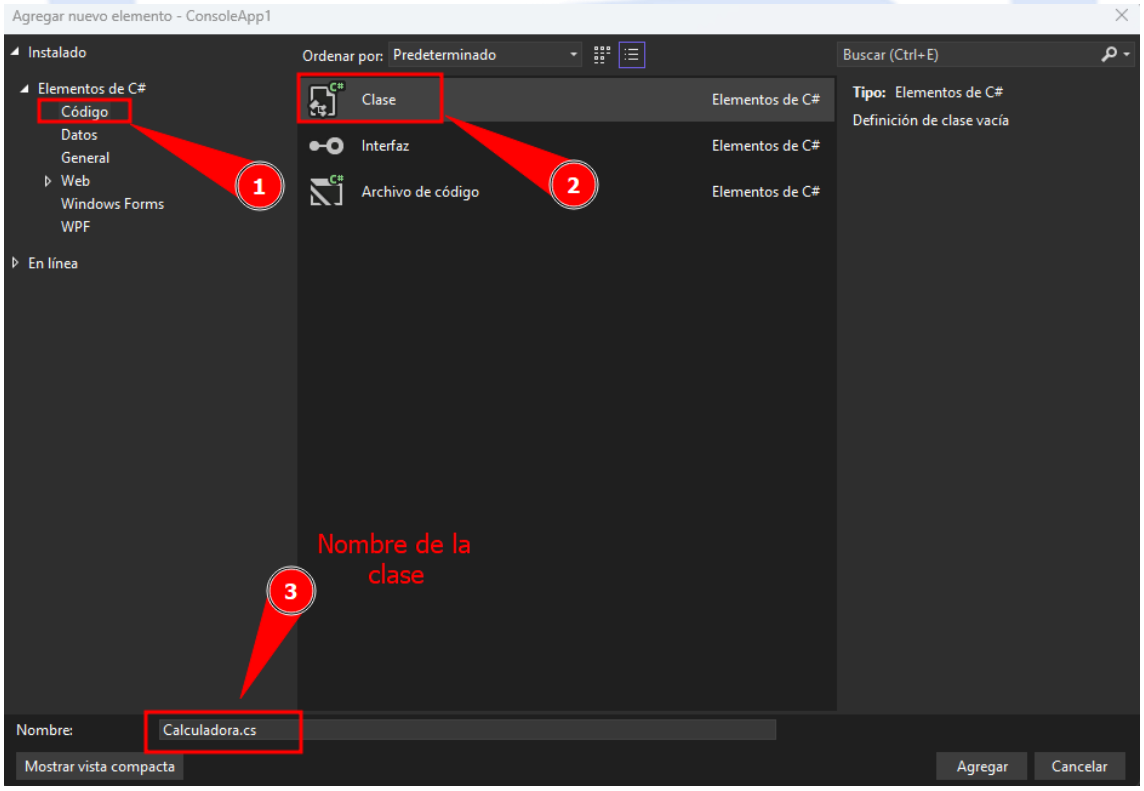




2.- Agregamos la clase “Calculadora”




Buscar en Explorador de soluciones (Ctrl+')
Solución "ConsoleApp1" (1 de 1 proyecto)
ConsoleApp1
Compilar
Recompilar
Limpiar
Análisis y limpieza de código
Publicar...
Actualizar
Contraer todos los descendientes Ctrl+Flecha izquierda
Limitar el ámbito a esto
Nueva vista de Explorador de soluciones
Agregar
Administrar paquetes NuGet...
Configurar proyectos de inicio...
Establecer como proyecto de inicio
Depurar
Inicializar el elemento interactivo con el proyecto
Cortar Ctrl+X
Pegar Ctrl+V
Quitar Supr
Cambiar nombre F2
Descargar el proyecto
Cargar dependencias directas
Cargar todo el árbol de dependencias
Copiar ruta de acceso completa
Abrir carpeta en el Explorador de archivos
Abrir en terminal
Propiedades Alt+Entrar
Nuevo elemento... Ctrl+Mayús.+A
Elemento existente... Mayús.+Alt+A
Nueva carpeta
Atención al cliente con .NET Aspire Orchestrator...
Cliente de API REST...
Modelo de Machine Learning...
Referencia...
Referencia web...
Referencia de servicio...
Servicio conectado
Analizador...
Formulario (Windows Forms)...
Control de usuario (Windows Forms)...
Componente...
Clase...
New EditorConfig
1
2


Agregar nuevo elemento - ConsoleApp1
Instalado
Elementos de C#
Código
Datos
General
Web
Windows Forms
WPF
En línea
Ordenar por: Predeterminado
Clase Elementos de C#
Interfaz Elementos de C#
Archivo de código Elementos de C#
Buscar (Ctrl+E)
Tipo: Elementos de C#
Definición de clase vacía
Nombre de la clase
Nombre: Calculadora.cs
Mostrar vista compacta
Agregar
Cancelar
1
2
3

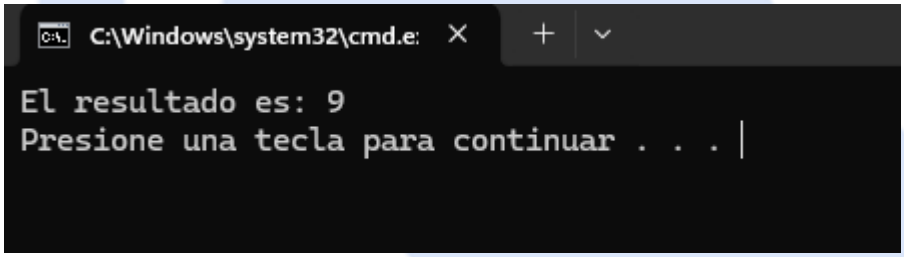
Paso 3.- escribimos el siguiente cogido para la clase “Calculadora”

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    2 referencias
    public class Calculadora
    {
        // Método privado para sumar dos números
        1 referencia
        private int Sumar(int a, int b)
        {
            return a + b;
        }

        // Método privado para mostrar el resultado internamente
        1 referencia
        private void MostrarResultado(int resultado)
        {
            Console.WriteLine("El resultado es: " + resultado);
        }

        // Método público que llama al método privado
        1 referencia
        public void CalcularYMostrar(int a, int b)
        {
            int resultado = Sumar(a, b);
            MostrarResultado(resultado);  // Llama al método privado
        }
    }
}
```

Paso 4.- Probamos el resultado llamando a la clase “Calculadora” y su método publico en el programa principal

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    0 referencias
    internal class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            Calculadora cal = new Calculadora();
            cal.CalcularYMostrar(2,7);
        }
    }
}
```

Resultado Esperado



Explicación:

- El método MostrarResultado es privado, solo puede ser llamado desde dentro de la clase **Calculadora**.
- El método CalcularYMostrar es público y llama al método privado MostrarResultado para imprimir el resultado de la suma en el programa principal.

Métodos que no retornan elementos (void)

Un método que no retorna ningún valor es del tipo void. Este tipo de métodos realizan una tarea específica, pero no devuelven un resultado.

EJEMPLO

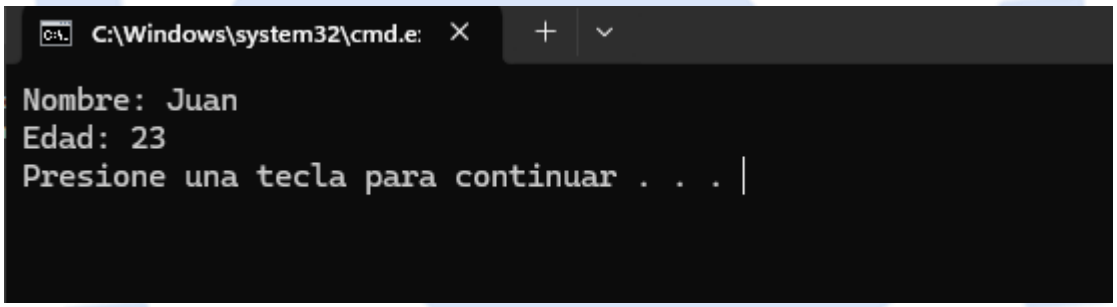
PASO 1.- Creamos la clase “Persona” y escribimos el siguiente código

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    3 referencias
    public class Persona
    {
        // Atributos privados
        private string nombre;
        private int edad;

        // Constructor para inicializar la persona
        1 referencia
        public Persona(string nombre, int edad)
        {
            this.nombre = nombre;
            this.edad = edad;
        }

        // Método público que muestra los detalles de la persona (tipo void)
        1 referencia
        public void MostrarDetalles()
        {
            Console.WriteLine("Nombre: " + nombre);
            Console.WriteLine("Edad: " + edad);
        }
    }
}
```

PASO 2.- Llamamos a la clase “Persona” y a su método publico en el programa principal

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    0 referencias
    internal class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            Persona persona = new Persona("Juan", 23);
            persona.MostrarDetalles();
        }
    }
}
```

Resultado esperado



Explicación:

El método MostrarDetalles no devuelve ningún valor; simplemente imprime los detalles de la persona en la consola. Esto lo hace un método de tipo void.

Combinación de private y void

Se puedes combinar ambos conceptos, es decir, un método privado de tipo void, que realiza una tarea auxiliar dentro de la clase.

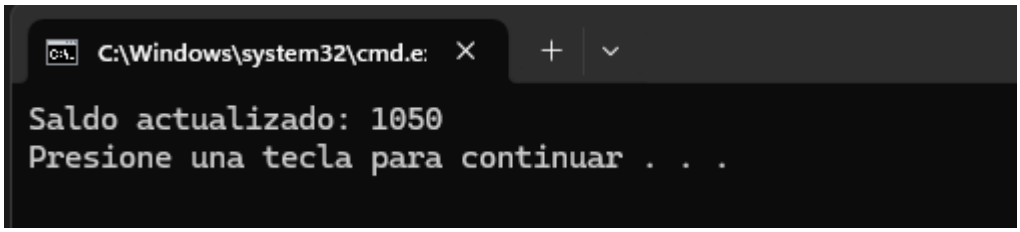
PASO 1.- Creamos la clase “Banco” y escribimos el siguiente código

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    3 referencias
    public class Banco
    {
        private double saldo;
        // Constructor
        1 referencia
        public Banco(double saldoInicial)
        {
            saldo = saldoInicial;
        }
        // Método público para depositar dinero
        1 referencia
        public void Depositar(double cantidad)
        {
            saldo += cantidad;
            ActualizarSaldo();  // Llamada a método privado
        }
        // Método privado tipo void, no retorna nada y actualiza el saldo
        1 referencia
        private void ActualizarSaldo()
        {
            Console.WriteLine("Saldo actualizado: " + saldo);
        }
    }
}
```

PASO 2.- Llamamos a la clase “Persona” y a su método público en el programa principal

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    0 referencias
    internal class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            Banco miCuenta = new Banco(1000);
            miCuenta.Depositar(50);
        }
    }
}
```

Resultado Esperado



Modificador public y métodos que retornan

Un tipo de dato

Los métodos con el modificador public y que retornan un tipo de dato son aquellos que pueden ser accedidos desde fuera de la clase y devuelven un valor específico cuando son llamados. El tipo de dato que retornan puede ser un tipo básico (como int, double, string, etc.) o un tipo complejo (como objetos, listas, etc.).

EJEMPLOS

Método que retorna un entero (int):

- Este método realiza una operación y devuelve un número entero como resultado.

1.- Modificamos el código de la clase “Calculadora”

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    0 referencias
    public class Calculadora
    {
        0 referencias
        public int Sumar(int a, int b)
        {
            return a + b;  // Retorna la suma de dos enteros
        }
    }
}
```

2.- Llamamos a la clase calculadora y su método público “Sumar” que contiene un valor de retorno tipo int.

```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    0 referencias
    internal class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            Calculadora calc = new Calculadora();
            int resultado = calc.Sumar(5, 3);
            Console.WriteLine("Resultado de la suma: " + resultado);
        }
    }
}
```

Método que retorna un valor booleano (bool):

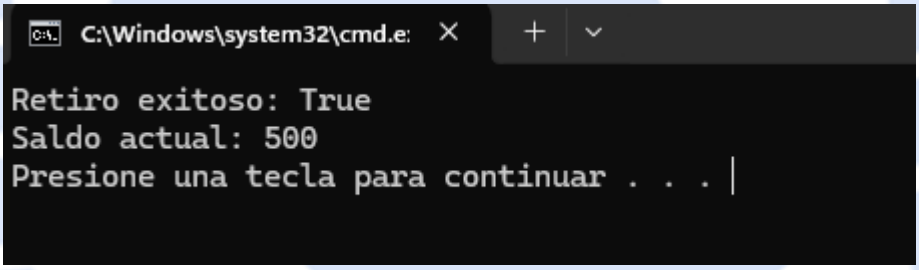
- Un método puede devolver un valor booleano (true o false) dependiendo de ciertas condiciones.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    3 referencias
    public class Banco
    {
        double saldo;
        //Constructor
        1 referencia
        public Banco(double saldoInicial)
        {
            this.saldo = saldoInicial;
        }
        //Método público que retorna un booleano
        1 referencia
        public bool Retirar(double cantidad)
        {
            if (cantidad <= saldo)
            {
                saldo -= cantidad;
                return true;  // Retorna true si la operación es exitosa
            }
            else
            {
                return false;  // Retorna false si no hay suficiente saldo
            }
        }
        // Método para consultar el saldo
        1 referencia
        public double ConsultarSaldo()
        {
            return saldo;
        }
    }
}
```

2.- Llamamos a la clase BANCO y su método público “Retirar y ConsultarSaldo” que contiene un valor de retorno tipo bool y double.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    0 referencias
    internal class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            Banco miCuenta = new Banco(1000);
            bool exito = miCuenta.Retirar(500);
            Console.WriteLine("Retiro exitoso: " + exito);  // Imprime: "Retiro exitoso: true o false"
            Console.WriteLine("Saldo actual: " + miCuenta.ConsultarSaldo());  // Imprime: "Saldo actual"
        }
    }
}
```

Resultado esperado



INSTITUTO

KHIPU