

---

# PROGRAMACIÓN WEB AVANZADA

---

## Actividad Práctica: Aplicación de Programación Orientada a Objetos.

### **Author**

Jairo Quilumbaquin

Sangolqui, Ecuador

Viernes 24 de Noviembre del 2023

# Contents

|          |                                   |          |
|----------|-----------------------------------|----------|
| <b>1</b> | <b>Objetivo</b>                   | <b>3</b> |
| 1.1      | Definición de Clases . . . . .    | 3        |
| 1.2      | Herencia . . . . .                | 5        |
| 1.3      | Interfaz y Polimorfismo . . . . . | 5        |
| 1.4      | Implementación . . . . .          | 5        |

# 1 Objetivo

Aplicar los conceptos de Programación Orientada a Objetos (POO) aprendidos en C# mediante la creación de una aplicación de gestión de empleados.

## 1.1 Definición de Clases

- Define una clase Empleado con los siguientes atributos.
  - nombre
  - salario
  - Implementa un método en la clase ‘Empleado’ para calcular el salario anual.

Implementación de la clase:

```
using System;

namespace HelloWorld
{
    class Empleado: IMostrarInformacion
    {
        protected string nombre;
        protected double salario;

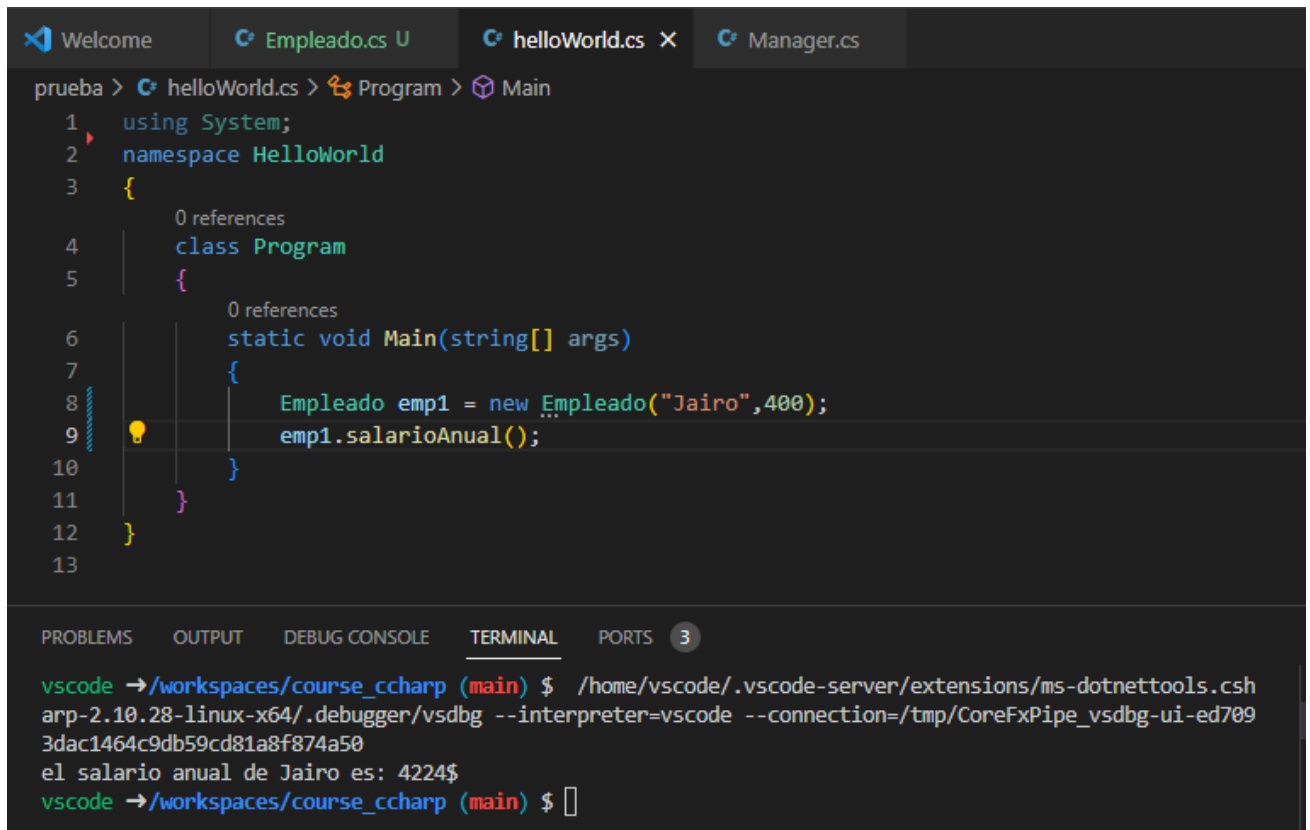
        public Empleado() {}
        public Empleado(string nombre, double salario) {
            this.nombre = nombre;
            this.salario = salario;
        }

        public void salarioAnual() {
            double total = (this.salario - this.salario*0.12)*12;
            Console.WriteLine($"el salario anual de {this.nombre} es: {total}$");
        }
        public void mostrarInformacion(){
            Console.WriteLine($"El nombre del empleado es de {this.nombre} y su salario es {this}");
        }

    }

}
```

Llamada del programa desde la clase principal en vscode:



The screenshot shows the Visual Studio Code interface with a C# project. The editor displays the following code in `helloWorld.cs`:

```
1 using System;
2 namespace HelloWorld
3 {
4     0 references
5     class Program
6     {
7         0 references
8         static void Main(string[] args)
9         {
10             Empleado emp1 = new Empleado("Jairo", 400);
11             emp1.salarioAnual();
12         }
13 }
```

The bottom panel shows the TERMINAL output:

```
vscode →/workspaces/course_csharp (main) $ /home/vscode/.vscode-server/extensions/ms-dotnettools.csharp-2.10.28-linux-x64/.debugger/vsdbg --interpreter=vscode --connection=/tmp/CoreFxPipe_vsdbg-ui-ed7093dac1464c9db59cd81a8f874a50
el salario anual de Jairo es: 4224$
vscode →/workspaces/course_csharp (main) $
```

Figure 1: llamada de la clase empleado en visual code usando dev containers

## 1.2 Herencia

- Crea una clase Gerente que herede de la clase Empleado.
- Agrega un nuevo atributo para el departamento que supervisa en la clase Gerente.

Implementación del código:

```
using System;
namespace HelloWorld
{
    class Manager:Empleado, IMostrarInformacion
    {
        string departamento;
        public Manager(){}
        public Manager(string nombre,double salario,string departamento):base(nombre,salario){
            this.departamento =departamento;
        }

        public void mostrarInformacion(){
            Console.WriteLine($"El nombre del manager es {nombre} y esta a cargo de {this.departamento}");
        }
    }
}
```

## 1.3 Interfaz y Polimorfismo

- Define una interfaz IMostrarInformacion con un método para mostrar información general.

Implementación del código:

```
using System;
namespace HelloWorld
{
    interface IMostrarInformacion
    {
        public void mostrarInformacion();
    }
}
```

## 1.4 Implementación

- Implementa la interfaz IMostrarInformacion en ambas clases (Empleado y Gerente).[1]
- Crea instancias de Empleado y Gerente y muestra su información utilizando el polimorfismo a través de la interfaz.

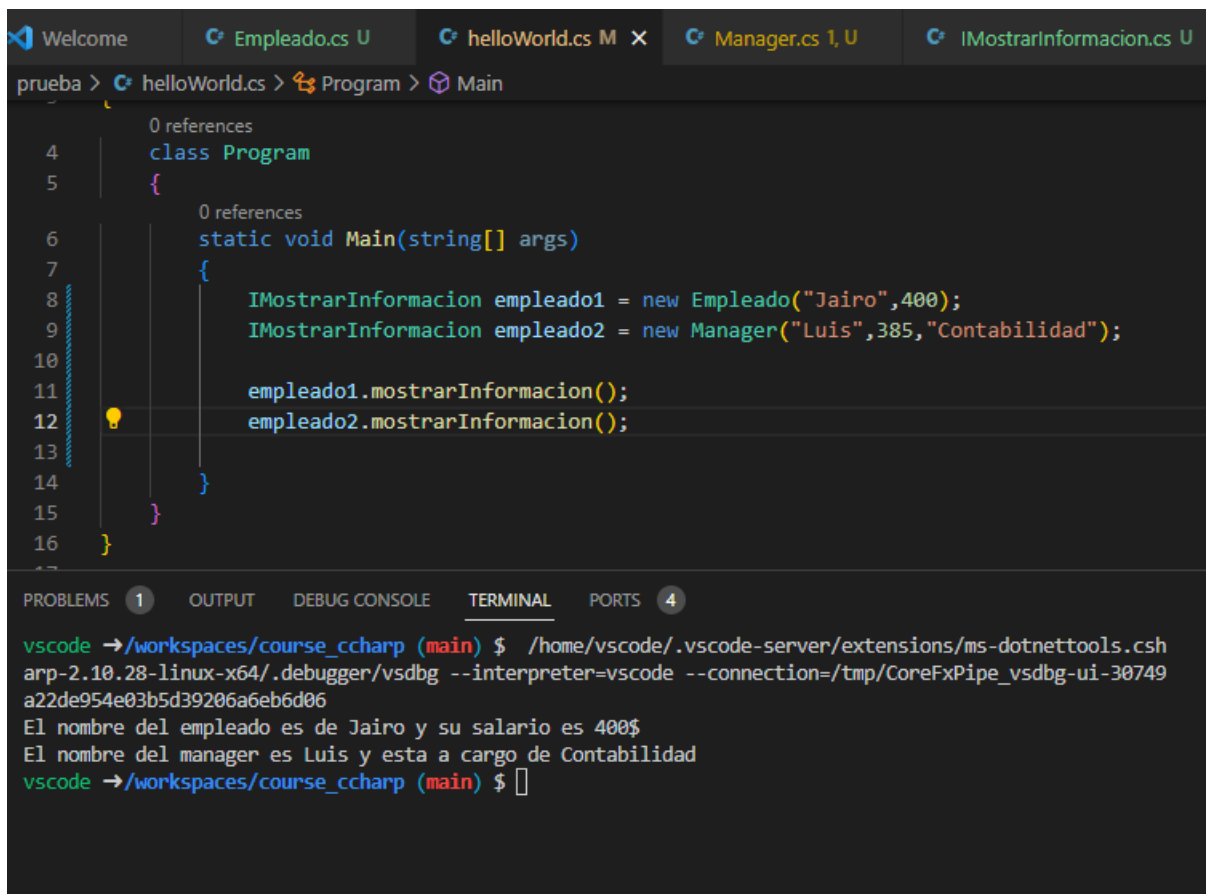
Código de la implementación:

```
using System;
namespace HelloWorld
{
    class Program
    {
        static void Main(string[] args)
        {
            IMostrarInformacion empleado1 = new Empleado("Jairo",400);
            IMostrarInformacion empleado2 = new Manager("Luis",385,"Contabilidad");

            empleado1.mostrarInformacion();
            empleado2.mostrarInformacion();

        }
    }
}
```

Resultado de la ejecución:



```
vscode → /workspaces/course_csharp (main) $ /home/vscode/.vscode-server/extensions/ms-dotnettools.csharp-2.10.28-linux-x64/.debugger/vsdbg --interpreter=vscode --connection=/tmp/CoreFxPipe_vsdbg-ui-30749-a22de954e03b5d39206a6eb6d06
El nombre del empleado es de Jairo y su salario es 400$
El nombre del manager es Luis y esta a cargo de Contabilidad
vscode → /workspaces/course_csharp (main) $
```

Figure 2: Ejecución del programa

## References

- [1] Microsoft. Tour de c# - estructura del programa. Recuperado el [fecha de acceso]. [Online]. Available: <https://learn.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/#program-structure>