ACTIVIDAD PRÁCTICA 9 (AP9)

Título

AP9. Objetos y Clases (POO)

Objetivos

- Realizar de forma correcta con lenguaje C# la declaración de Clases, la creación de atributos (información y datos) pertenecientes a la clase, la creación de métodos (comportamiento) dentro de la clase.
- Saber instanciar objetos de una clase y la invocación de métodos del objeto.
- Utilizar correctamente los ámbitos a nivel local, nivel de objeto y a nivel de Clase.
- Utilizar los métodos de acceso público <public> y privado <private> de atributos y métodos.
- Utilizar métodos o funciones de acceso (accesors) – Funciones de interfaces tipo Getter y Setter.
- Utilizar Propiedades para el acceso de lectura y escritura a los atributos privados del objeto.
- Utilizar Constructores para inicializar la información de un objeto.
- Utilizar la sobrecarga de constructores y de métodos.
- Diferencias entre la parte de interface e implementación de una clase/objeto.

Temporalización

Previsto: Utilización de una sesión presenciales + trabajo en casa.

Proceso de desarrollo

Realiza los siguientes ejercicios prácticos propuestos.

*Debes de realizar la entrega de todos los proyectos/ejercicios propuestos adjuntándolos en un archivo comprimido (**AP9.zip** o **AP9.rar**). Cada proyecto deberá estar en una carpeta diferente (EJ01, EJ02, etc.)*

EJERCICIO 1.

Desarrolla una aplicación con una clase llamada **Socio** que tenga las siguientes condiciones:

- Sus atributos son: nombre, edad, NIF, sexo (H - hombre, M - mujer), peso y altura. Si quieres añadir algún atributo de forma adicional puedes realizarlo.
- Desarrolla un constructor que reciba el nombre, la edad, sexo, peso y altura y otro constructor sobrecargado que pida los anteriores datos por teclado y se los asigne a los atributos del objeto.
- Implementa los siguientes métodos que formarán parte de la interface de la Clase:
  - **CalcularIMC()**: El índice de masa corporal calculará si la persona está en su peso ideal (la fórmula es IMC = peso en kg/(altura^2 en metros)). Si esta fórmula devuelve un valor menor que 18,5 indica que tenemos un peso bajo, si es mayor o igual que 18,5 y menor que 25 es que estamos en el peso normal, y si es 25 o mayor es que tenemos sobrepeso. El método devolverá/retornará un string que indique el estado de peso ("bajo", "normal", "alto").
  - **EsMayorDeEdad()**: indica si es mayor de edad, devuelve un booleano (True o False)
  - **Informa():** visualiza toda la información del objeto en un formato narrativo. Un ejemplo posible sería:

```
Nombre: SALVA    Dni: 16454749C
Edad: 58         Sexo: hombre
Peso: 80         Altura: 1.81
```

- **GeneraNIF()**: genera un número aleatorio de 8 cifras, y a partir del mismo, genera su número de letra correspondiente para crear el NIF. Este método será invocado cuando se construya o instancie el objeto. Este método no formará parte de la interface del objeto, siendo parte de su implementación (deberá tener un acceso privado -sólo accesible desde el objeto-).

Al ejecutar el programa (en la clase Program):

1) Se deberán crear 2 objetos de la clase Socio (s1 y s2), instanciados cada uno con un constructor diferente de los disponibles y creados en la clase.

2) Posteriormente deberás presentar y visualizar la información de los dos socios: nombre, edad, sexo, altura, NIF, su estado de peso (bajo, normal o alto) y si es mayor de edad o no. Haz uso de las funciones de interface creadas para la clase.

Orientación para la visualización

```
Información del primer socio:
=============================

Introduce el nombre del Socio: Andrés
Introduce la edad del Socio: 56
Introduce el sexo del Socio (H/M): H
Introduce el peso del Socio: 92,5
Introduce la altura del Socio: 1,72

Nombre: ANDRÉS  DNI: 81718966N
Edad: 56        Sexo:H
Peso: 92,5      Altura:1,72

ANDRÉS tiene un peso alto
ANDRÉS es mayor de edad

Información del segundo socio:
==============================

Nombre: JUAN    DNI: 43112286M
Edad: 30        Sexo:H
Peso: 80,5      Altura:1,6

JUAN tiene un peso alto
JUAN es mayor de edad
```

EJERCICIO 2.

Construye una clase en C# que represente un triángulo. La clase debe incluir los siguientes métodos que devuelven un valor lógico indicando el tipo de triángulo:

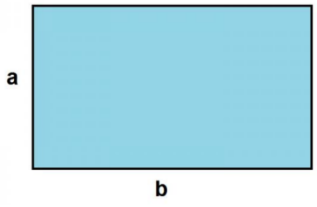
- EsEscaleno → todos los lados distintos.
- EsIsósceles → dos lados iguales y el otro distinto
- EsEquilatero → los tres lados iguales

Crea dos Constructores para poder dar valor a los lados del triángulo. Un primer constructor que reciba los datos como parámetros y un segundo constructor (sobrecargado) que pida los datos de los lados por teclado.

Prueba tu clase Triangulo creando desde el método Main de la clase Program, varios objetos de tipo triángulo (con los constructores implementados) y visualiza si son escalenos, isósceles o equiláteros.

EJERCICIO 3.

Construye un programa que calcule la superficie y el perímetro de un cuadrilátero (cuadrado o rectángulo) dada la longitud de sus dos lados. Los valores de los lados deberán ser suministrados al Constructor mediante parámetros. Para crear el objeto se deberá permitir suministrar un solo lado para los casos en que el cuadrilátero sea un cuadrado.



$A = ba$

$P = 2b + 2a$

EJERCICIO 4.

Construye un programa en C# que, utilizando un menú de opciones, permita dirigir el movimiento de un objeto dentro de un tablero y actualice su posición dentro del mismo. Los movimientos posibles son ARRIBA, ABAJO, IZQUIERDA, DERECHA. Tras cada movimiento el programa mostrará la nueva dirección elegida y las coordenadas de situación del objeto dentro del tablero. Las coordenadas o posiciones iniciales serán (5,5) y en todo momento en movimiento sobre el tablero deberá realizarse sobre coordenadas positivas (mayores a cero). En caso de llegar a situarse en posiciones cero (para X o Y), se deberá mantener la posición en 1.

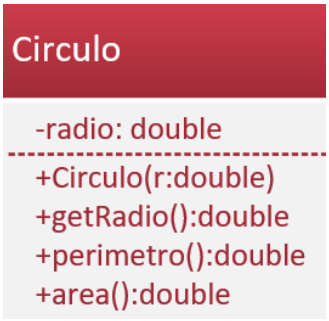
```
Movimientos permitidos
======================
1. Arriba.
2. Abajo.
3. Izquierda.
4. Derecha.
5. Salir / Acabar.
Introduce Opción (1..5): 1

Coord.X = 5 -- Coord.Y = 6
<Pulsa tecla>
```

EJERCICIO 5.

Construye un programa en C# que, dados una serie de vehículos caracterizados por su marca, modelo, potencia y precio imprima las propiedades e información del vehículo más barato. Para ello, al ejecutarse el programa se deberán pedir por teclado el número de vehículos a considerar y posteriormente leer por teclado las características de cada vehículo.

EJERCICIO 6.

Construye un programa en C# que implemente una clase para trabajar con Círculos. Para ello, definir las variables de estado mínimas que requiera, proporcionando métodos de consulta, un método constructor y métodos para calcular el perímetro y el área de un circulo. Crea una propiedad Radio para poder acceder/leer la información del radio del círculo.



A modo de orientación se suministra un diagrama UML de la clase aproximada.

Además, al ejecutar el programa crea 2 círculos (uno de radio 5 y otro de radio 12) y visualiza el valor de su perímetro y área.

Por último, en la clase Program, implementa un método que a partir de un array de círculos que reciba como parámetro, devuelva el de mayor diámetro.

Evaluación

Esta actividad práctica no es una actividad de evaluación. Su realización forma parte del 10% de evaluación correspondiente al seguimiento e interés de la asignatura por parte del alumnado. Supone un refuerzo / estudio de cara a la realización de las actividades de evaluación entregables y las pruebas teórico-prácticas.

Recursos

Disponibles en plataforma (Recursos didácticos del Bloque3).