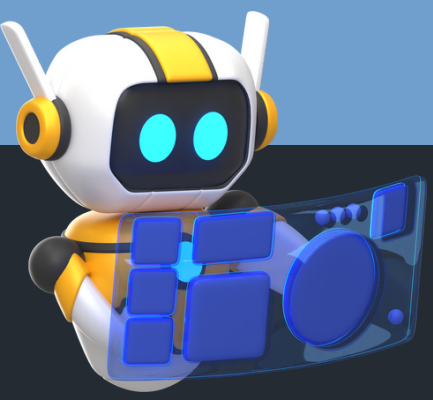
ROSEMBERG ESCOBAR -1046925



CLASES

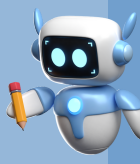
DISEÑO Y USO DE CLASES

En POO, las clases son bloques de creación de una aplicación.

- **Clases**: Planos técnicos que se usan para definir datos o propiedades y métodos de los objetos.
- **Objetos**: Instancias de clases, cuando se ccrea una instancia de una clase, se crea un objeto de memoria.

Los objetos tienen atributos tanto físicos como de comportamiento.

- **Físicos**: Aspectos del objeto que se puede ver o medir.
- **Comportamiento**: Aspectos de capacidad o interacció con otros objetos.



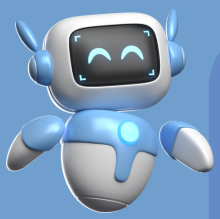
Lapiz

Físicos: Propiedades
Longitud, color, nitidez, borrador.

Comportamiento: Métodos
Capacidad de escribir, borrar y afilar.

Los espacios de nombre (namespace) son útiles para organizar muchas clases y controlar ámbtos de los nombre de clase.
Al utilizar nombres completos, se puede perder la legibilidad del código, para ello se utiliza using, que ayuda a que no se tenga que especificar el namespace.

Las propiedades y métodos se definen en una clase, esta clase es la plantilla para decirle que hacer, a la se le llama definición de clase.



Conceptos fundamentales:

- ***Abstracción***: Proceso de simplificar y representar una entidad.
- ***Encapsulamiento***: Proteger los datos internos del objeto.
- ***Herencia***: Permite ccrear nuevas clases basadas en clases existentes.
- ***Polimorfismo***: Permitir que método con el mismo nombre tengan diferentes comportamientos.

CONSTRUCTORES DE CLASES

Un constructor de una clase es un método que inicializa un objeto cuando se crea una instancia.

- Constructor ***instancia***: Se usan para crear einicializar cualquier variable del campo.
- Constructor ***estático***: Se usan para inicializar datos estáticos o valores determinados.

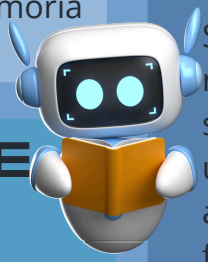
EJEMPLO

```
class Coche(){
      int numeroDeRuedas;

      Coche(){
            this.numeroDeRuedas = 4;
       }
      Coche(int numeroDeRuedas){
            this.numeroDeRuedas = numeroDeRuedas;
       }

}
```

CONSTRUCTOR DE INSTANCIA

No tienen tipo de retorno (ni void), tienen el mismo nombre que su clase, se ejecutan automáticamente al instanciar, pueden tener modificadores de acceso y pueden sobrecargar para crear diferentes formas.
Para instanciar objetos se utiliza la palabra reservada ***new***. Esto hace que se llame al constructor y genere el bloque de memoria del objeto.

La ***sobrecarga de constructores*** se refiere cuando existen más de un constructor que tiene el mismo nombre, este permite tenerlos con diferentes parámetros.



CONSTRUCTOR ESTÁTICO

Se usa para inicializar datos estáticos o realizar una acción determinada, que solo se hace una vez, se llama antes de hacer una istancia. No tienen modificadores de acceso, no tiene parámetros, de igual forma, el usuario no tiene control cuando se ejecuta.

PALABRAS CLAVE

Para el acceso de las propiedades que permiten controlar los datos de una clase. Tienen dos partes:

- ***Get***: Solo lectura.
- ***Set***: Asigna valor a una propiedad.

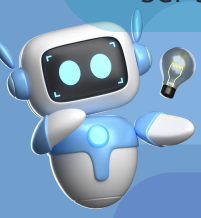
Palabras clave del lenguaje para las clases:

- ***class***: Define una clase.
- ***new***: Crear una nueva instancia de una clase.
- ***abstract***: Evitar que una clase pueda ser instanciada.
- ***base***: Invocar el construcor de la clase padre.
- ***virtual***: Permite que un método pueda ser sobreescrito en clases dereivadas.

ACCESOS

Los modificadores de acceso son permisos que controlan quien puede ver o cambiar partes del código. Con estos se puede controlar quien cambia o mira códigom evitar errores de codificación, evitar vulnerabilidades. Estos son:

- ***public***: cualquiera puede acceder.
- ***private***: solo la misma clase puede acceder.
- ***protected***: solo la clase y sus clases derivadas acceden.
- ***internal***: solo el código dentro del mismo proyecto puede acceder.



PARÁMETROS

Cuando se crean métodos, se deben de definir su firma. Se define por nivel de acceso y su tipo de retorno o en modificadores en él el nombre y el parámetro. Tienen dos diferentes tipos:

- **Por valor**: Cuando es un tipo valor, se envia una copia del objeto al método, los cambios realizados en el método no cambian al original. Cuando es de referencia se pasa a una dirección del objeto, donde sí se puede cambiar.
- **Por referencia:** Al utilizar ref se puede modificar la variable original.