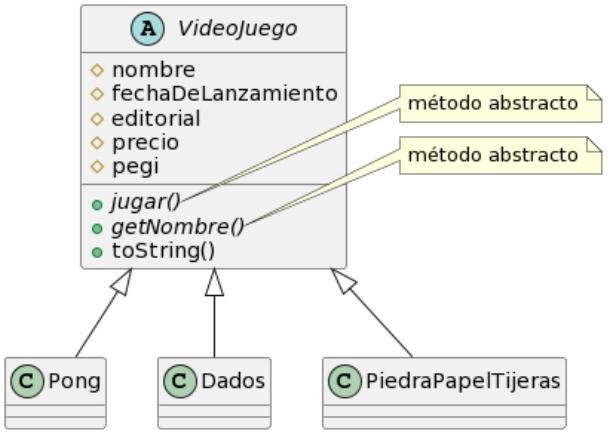
PRACTICO 5

Clases Concretas, Abstractas, Interfaces.
Modificador Final

Ej. 5.1

Dado el siguiente diagrama de clases, implementar el codigo Java que lo represente:



El constructor de cada clase debe imprimir un mensaje indicando el tipo de objeto que se crea. Por ejemplo, la clase Pong, imprimiría "*Se crea un objeto de tipo Pong*".
En el método *main* cree instancias de las diferentes clases concretas y observe los resultados en la salida estándar.

Ej. 5.2

Declare una interfaz llamada ***Multiplicable*** con los siguientes métodos: ***void multiplicar(int n)*** y ***void mostrarResultado***(). Luego cree las siguientes clases concretas, que implementen la interfaz ***Multiplicable,*** con sus correspondientes atributos:

- Clase ***NumeroEntero*** con atributo ***int valor;***
- Clase ***MiVector*** *con atributo* ***int[] valor;*** y
- Clase **Mi*Matriz*** con atributo ***int[][] valor***;

La implementación de ***multiplicar(int n)*** de cada clase debe modificar los atributos propios.

Defina y cree un ArrayList de objetos Multiplicable (ArrayList<Multiplicable>) que contenga instancias de cada clase concreta. Recorra esta estructura invocando los métodos necesarios para multiplicar por un número dado y mostrar los resultados.

Ej. 5.3

Reusar la clase **VideoJuego**, definir un método concreto m*ostrarResultado()* que imprima por salida estándar el contenido de un atributo **resultado** (de tipo *String*) de esa misma clase. Rápidamente dentro de un *main(...)* intentar instanciar un objeto de clase **VideoJuego**. Analizar lo ocurrido.

Crear dos subclases concretas de **VideoJuego**:

- **Dados**, que implemente el método *jugar()* para que genere un número aleatorio (entre 1 y 6) y lo guarde en resultado.
- **PiedraPapelTijera**, que implemente el método *jugar()* para que genere aleatoriamente un resultado “piedra”, "papel" o “tijera” y lo guarde en resultado.

Implementar en el método *main(...)* la creación y uso de estos juegos, almacenando las instancias en un vector de Juegos, mostrando, en cada caso, el resultado de *jugar()*.

Ej. 5.4

a) **Diseñe en UML** un sistema para administrar las cuentas de un banco. Una cuenta soporta las siguientes operaciones:

- depositar un monto determinado,
- extraer un monto determinado,
- consultar el saldo de la cuenta,
- transferir un monto desde esta cuenta a otra,

El sistema debe permitir el uso de cuentas de caja de ahorro y cuentas corrientes. Toda cuenta debe conocer número de cuenta, titular y saldo.
Las operaciones de depósito, extracción y consulta de saldo no deben ser implementadas en la clase *Cuenta* sin embargo deben poder ser implementadas por posibles subclases. La operación de transferencia debe usar las operaciones de depósito y extracción para implementarla. No debe ser posible alterar esta operación por ninguna subclase, pero ellas pueden hacer uso de esta operación. Las operaciones de depósito y extracción deben ser accesibles desde cualquier clase. La operación de consulta de saldo sólo debe estar disponible desde las subclases. Adicionalmente el sistema debe incluir Clientes que poseen cuentas bancarias. El cliente posee nombre, apellido y dirección. Además, para una cuenta debe poder consultarse su titular (el cliente).

b) **Implemente en Java** un sistema que permita administrar clientes y sus cuentas bancarias, según lo diseñado en el inciso anterior.

c) Modificar la clase cuenta de manera que genere en forma automática y correlativa los números de cuenta para las nuevas cuentas, usando un **atributo de clase y un método de clase**.