Ejercicio 1

Desarrolla una aplicación que gestione jugadores de eSports en distintos videojuegos. Para ello vas a crear una clase base llamada Jugador con los siguientes atributos: Nombre del jugador, nacionalidad, nombre del equipo, cantidad de torneos ganados, puntos en el ranking y la categoría en la que está. De esta clase salen cuatro subclases con un atributo propio cada uno, son las siguientes:

- JugadorLoL: rol principal (Top, Mid, Jungla, ADC, Support).
- JugadorCSGO: precisión en porcentaje (%).
- JugadorFIFA: número de goles anotados en torneos.
- JugadorValorant: agente favorito utilizado.

Por otro lado vas a tener que desarrollar los siguientes métodos:

- ganarTorneo(puntos): incrementa los puntos del ranking al ganar un torneo.
- subirNivel(): sube de categoría dentro del mismo videojuego, cambiando a una liga más competitiva.

Junto a esto vas a tener que desarrollar todos los constructores, los getters y setters y un método main que cree una instancia de cada tipo (con datos que el usuario mete por el teclado) y llame a cada método de uno en uno. Como parte opcional podéis crear los JUnits.

Ejercicio 2

Desarrolla una aplicación que gestione atletas olímpicos. Para ello vas a crear una clase base llamada Atleta con los siguientes atributos: Nombre, nacionalidad, edad, disciplina y medallas ganadas. De esta clase salen cuatro subclases con un atributo propio cada uno, son las siguientes:

- AtletaNatacion: estilo principal (libre, espalda, mariposa, pecho).
- AtletaAtletismo: distancia preferida (100m, 200m, maratón).
- AtletaGimnasia: tipo de aparato usado (barra, suelo, anillas).
- AtletaCiclismo: tipo de bicicleta usada (ruta, montaña, pista).

Por otro lado vas a tener que desarrollar los siguientes métodos:

- ganarMedalla(tipo): añade una medalla (oro, plata, bronce) al contador.
- mejorarMarca(nueva_marca): actualiza la mejor marca personal del atleta.
- cambiarDisciplina(nueva_disciplina): permite que el atleta se especialice en otra disciplina similar.

Junto a esto vas a tener que desarrollar todos los constructores, los getters y setters y un método main que cree una instancia de cada tipo (con datos que el usuario mete por el teclado) y llame a cada método de uno en uno. Como parte opcional podéis crear los JUnits.

Ejercicio 3

Desarrolla una aplicación que gestione aeronaves. Para ello vas a crear una clase base llamada Aeronave con los siguientes atributos: Fabricante, modelo, capacidad de pasajeros, autonomía de vuelo (en km) y velocidad máxima (en km/h). De esta clase salen cuatro subclases con un atributo propio cada uno, son las siguientes:

- AvionComercial: número de aerolínea y cantidad de clases (turista, business, primera clase).
- AvionCarga: capacidad de carga en toneladas y tipo de mercancía principal.
- CazaMilitar: tipo de armamento principal y velocidad de ascenso (m/s).
- Helicoptero: número de rotores y si es de uso civil o militar.

Por otro lado vas a tener que desarrollar los siguientes métodos:

- calcularTiempoVuelo(distancia): devuelve el tiempo estimado de vuelo según la autonomía.
- repostarCombustible(): simula el reabastecimiento de combustible y reajusta la autonomía.
- actualizarModelo(nuevo_modelo): cambia el modelo de la aeronave si se actualiza su versión.

Junto a esto vas a tener que desarrollar todos los constructores, los getters y setters y un método main que cree una instancia de cada tipo (con datos que el usuario mete por el teclado) y llame a cada método de uno en uno. Como parte opcional podéis crear los JUnits.

Ejercicio 4

Desarrolla una aplicación que gestione el personal de un hospital. Para ello vas a crear una clase base llamada Personal con los siguientes atributos: Nombre, edad, número de identificación, el turno de trabajo (mañana, tarde y noche) y los años de experiencia. De esta clase salen cuatro subclases con un atributo propio cada uno, son las siguientes:

- Doctor: especialidad médica y número de pacientes atendidos.
- Enfermero: área asignada (urgencias, pediatría, quirófano) y cantidad de pacientes bajo su cuidado.
- Cirujano: cantidad de cirugías realizadas y tipo de cirugías que puede realizar.
- Administrativo: departamento en el que trabaja y si tiene acceso a datos confidenciales.

Por otro lado vas a tener que desarrollar los siguientes métodos:

- atenderPaciente(): imprime un mensaje indicando que el profesional ha atendido a un paciente y suma uno al contador.
- cambiarTurno(nuevo_turno): actualiza el turno de trabajo del empleado.
- ascenderPuesto(): permite a un empleado ascender dentro del hospital si cumple ciertos requisitos (por ejemplo, un enfermero puede ascender a supervisor).

Junto a esto vas a tener que desarrollar todos los constructores, los getters y setters y un método main que cree una instancia de cada tipo (con datos que el usuario mete por el teclado) y llame a cada método de uno en uno. Como parte opcional podéis crear los JUnits.