UNIVERSIDAD AUTÓNOMA "TOMÁS FRÍAS" INGENIERÍA DE SISTEMAS				
ESTUDIANTE:			**C.I.:**	
DOCENTE:	Ing. Ditmar Castro Angulo	**MATERIA:**	SIS-211 G2	**PRÁCTICA N3 AUX**
AUXILIAR:	Univ. Gabriel Alejandro Garvizu Salas	**FECHA:**	11/10/2024	

Realizar un solo programa que cumpla con cada uno de los siguientes ejercicios sobre herencia

Ejercicio 1: Clase "Personaje"

Crea una clase llamada Personaje para un juego. Cada personaje tiene las siguientes características:

- Nombre (String)
- Nivel (int)
- Puntos de vida (int)

Y los siguientes métodos:

- mostrarEstado(): Imprime el estado actual del personaje (nombre, nivel y puntos de vida).
- recibirDaño(int daño): Reduce los puntos de vida del personaje basado en el daño recibido.
- curar(): Aumenta los puntos de vida del personaje en 20 puntos.

Ejercicio 2: Clase "Inventario"

Crea una clase llamada Inventario que permita a un personaje llevar un conjunto de ítems. La clase debe tener las siguientes características:

- Un arreglo o lista de Strings para almacenar los nombres de los ítems.
- Un método agregarItem(String item) que añada un ítem al inventario.
- Un método mostrarItems() que imprima todos los ítems en el inventario.

Debe de usarse esta clase dentro del constructor de la clase personaje

Ejercicio 3: Clase "Mago", "Arquero" y "Guerrero" como Subclase

Extiende la clase "Personaje" para crear tres tipos específicos de personajes: "Mago", "Arquero" y "Guerrero". Cada uno tiene características especiales:

- "Mago" tiene un atributo adicional llamado "mana" (int) y un método "lanzarHechizo()" que imprime un mensaje indicando que el mago ha lanzado un hechizo.
- "Arquero" tiene un atributo adicional llamado "destreza" (int) y un método "dispararFlecha()" que imprime un mensaje indicando que el arquero ha disparado una flecha.
- "Guerrero" tiene un atributo adicional llamado "fuerza" (int) y un método "atacar()" que imprime un mensaje indicando que el guerrero ha realizado un ataque.

Ejercicio 4: Juego simple de combate

Implementa un juego simple de combate en el Main donde creas dos personajes, un mago y un guerrero. Ambos comienzan con 100 puntos de vida y niveles aleatorios. El juego consiste en simular un combate donde:

- Cada personaje ataca al otro en turno (el mago usando lanzarHechizo() y el guerrero atacar()), aplicando daño aleatorio entre 10 y 20 puntos.
- Después de cada ataque, muestra el estado de ambos personajes.
- El juego termina cuando uno de los personajes se queda sin puntos de vida.

Ejercicio 5: Clase "Enemigo"

Crea una clase base llamada Enemigo. Los enemigos en un juego tienen los siguientes atributos:

- Nombre (String)
- Puntos de vida (int)
- Daño base (int)

Y los métodos:

- atacar(): Devuelve un valor de daño entero que inflige el enemigo.
- recibirDaño(int daño): Disminuye los puntos de vida del enemigo según el daño recibido.
- estaVivo(): Devuelve un booleano indicando si el enemigo aún tiene puntos de vida.

A partir de esta clase base, desarrolla dos clases derivadas:

- "Zombie", que tiene la característica especial de regenerar 5 puntos de vida cada vez que ataca.
- "Vampiro", que tiene la capacidad de robar vida (la mitad del daño infligido) al atacar.

Ejercicio 6: Clase "Jefe" como Subclase

Implementa una clase Jefe que herede de Enemigo y añade los siguientes atributos y comportamientos:

- Multiplicador de daño (double)
- hablar(): Un método que permite al jefe decir una frase amenazadora antes de comenzar el combate.
- La clase Jefe debe tener un método ataqueCritico que considere el multiplicador de daño para calcular el daño infligido, lo que lo hace más peligroso que un enemigo común.

Ejercicio 7: Implementación de un Mini Juego

Utiliza todas las clases y subclases anteriores para crear un mini juego donde puedas interactuar con diferentes personajes y/o combatir enemigos. La lógica del juego debe permitir que el jugador elija acciones como atacar, hablar o realizar alguna otra acción que tenga implementada

Ejercicio 8

Una vez completados los 7 ejercicios anteriores, es hora de realizar el ejercicio que vale más puntaje, hasta se podría decir que este ejercicio vale 100% de la práctica, puesto que si la carpeta con los distintos archivos no es subida a su repo de GitHub no podrá calificarse. Talvez ya te hayas percatado de que va, en caso de que no, ahí va.

Deberá de subirse los distintos archivos que se desarrollaron a lo largo de la práctica, a su repo en GitHub para su revisión, caso contrario no será válida la entrega por ningún otro medio.

Buena suerte y nos vemos en la siguiente práctica.

Fecha de entrega límite: 11/10/2024