{Ejercicio 4. Modelación con herencia}

Se subió otra tarea por equivocación. Esta entrega es un avance, solo para que aparezca en lugar de la tarea que no corresponde.

# Análisis del Sistema (30 puntos)

## Requisitos funcionales del sistema (5 pts)

Escribir los puntos exactos que debe cumplir tu sistema.

Ejemplo: “El usuario puede crear una nueva solicitud de boletos”, etc.

- Se debe mantener un control de turnos y rondas.

- Se debe mantener un control de atributos de todos los combatientes.

- Se debe ser capaz de diferenciar entre roles para los jugadores y tipos monstruos para los enemigos.

- Se deben poder cambiar los atributos mediante objetos y ataques.

- Se debe poder escoger el enemigo que se quiere atacar.

## Clases necesarias y su propósito (5 pts)

|  |  |
| --- | --- |
| Clase | Propósito |
| Main | Clase de instancia principal. |
| Consola | Interacción con el usuario. |
| Game | Clase que contendrá la lógica y los elementos del juego |
| Combatant | Clase padre de jugador y enemigo que contendrá la información de atributos generales. |
| Player | Clase padre que contará con información general del jugador. |
| Warrior | Clase hija de jugador con información del rol guerrero. |
| Explorer | Clase hija de jugador con información del rol explorador. |
| Enemy | Clase padre que contará con información general de los enemigos. |
| Goblin | Clase hija de enemigo con general información del duende. |
| Skeleton | Clase hija de enemigo con general información del esqueleto. |
| GoblinNormal | Clase hija de Goblin con información de un goblin con estadísticas normales. |
| GoblinBoss | Clase hija de Goblin con información de un goblin con estadísticas mejoradas. |
| SkeletonNormal | Clase hija de Skeleton con información de un esqueleto con estadísticas normales. |
| SkeletonBoss | Clase hija de Skeleton con información de un esqueleto con estadísticas mejoradas. |
| Item | Clase con información de cantidad, espacio que ocupa y efectos de ítems. |
| Bag | Clase con información de los ítems de cada jugador. |
| Attack | Clase con información y efectos de los ataques. |
| Ability | Clase con información de efecto y disponibilidad de las habilidades. |
| Shop | Clase con información de todos los objetos. Se usará para asignar objetos al inicio de la partida. |

* 1. **Atributos de cada clase (10 pts)**

Game:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| jugadores  enemigos  objetos | ArrayList <Combatant>  ArrayList <Combatant>  Shop | Private  Private  Private | Control de personajes en el juego.  Objetos disponibles |

Combatant:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Nombre  ataque  defensa  vida | String  int  int  int | public  public  public  public | Nombre del personaje  Ataque de personaje.  Defensa de personaje.  Vida de personaje |

Player:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| ataques  bolsa  espacioBolsa | Attack[]  Bag  int | public  public  public | Ataques del jugador  Items que tiene el jugador  Máximo de objetos |

Warrior, Explorer:

Atributos de Player adaptados por rol.

Enemy:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| habilidad  ataque | Ability  Attack | public  public | Habilidad del enemigo  Ataque del enemigo |

GoblinNormal, GoblinBoss, SkeletonNormal, SkeletoBoss:

Atributos de Enemy adaptados.

Item:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| general  cantidad  espacio | Ability  int  int | public  public  public | Efectos generales de habilidad  Cuantos se tiene  Espacio que ocupa en la bolsa |

Bag:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| objetos  espacio  disponible | ArrayList<Item>  int  int | public  public  public | Objetos guardados  Espacio de la bolsa  Espacio no utilizado. |

Attack:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| nombre  efectoAtaque  efectoDefensa  efectoVida | String  int  int  int | public  public  public  public | Identificador  Efectos que ocasiona |

Ability:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| nombre  efectoAtaque  efectoDefensa  efectoVida | String  int  int  int | public  public  public  public | Identificador  Efectos que ocasiona |

Shop:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| objetos | Item[] | public | Objetos guardados |

## Métodos de cada clase (10 pts)

Variables tendrán setter y getter: para los integer tendrán modificadores, mientras que los ArrayList tendrán adder y removers en lugar de setters.

Combatant:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atributo | Método | Parámetros : Tipo de dato | Visibilidad | Propósito |
| Attack  Ability | Atacar  UsarAbilidad | (int índice)  (int índice) | Public  Public | Serán heredadas y modificadas para jugador y enemigo. |

Game:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atributo | Método | Parámetros : Tipo de dato | Visibilidad | Propósito |
| void  String | inicioPartida  Ganador | ()  () | Public  Public | Maneja el juego.  Muestra ganador. (Jugadores / Enemigos) |

1. **Diseño: Diagrama de Clases (30 puntos)**
   * Asegúrate de mostrar atributos y métodos con visibilidad (+, -).
   * Indica relaciones entre clases (asociación, agregación, etc.).
   * Incluye el driver program (Main).

<https://lucid.app/lucidchart/ea0fbf4d-b80f-4346-8f39-f0c984df9e16/edit?viewport_loc=-310%2C-77%2C4253%2C1562%2CHWEp-vi-RSFO&invitationId=inv_3898b4dd-f7d8-4d06-b29b-1dbbb532ead4>

A diagram of a game

AI-generated content may be incorrect.

# Programa (40 puntos)

En cada archivo `.java`, asegurarse de incluir:

* + Las clases necesarias.
  + Uso adecuado de objetos.

Menú que debe implementar el driver program:

* + 1. Nuevo comprador
    2. Nueva solicitud de boletos
    3. …
    4. …
    5. Salir

GitHub: colocar aquí la URL: <https://github.com/Hola2212/Ejercicio4_Modelacion_con_herencia.git>

# Checklist antes de entregar

⬛✓⬛ Está claro el análisis?

⬛✓⬛ El diagrama tiene los elementos UML correctamente?

⬛✓ Subiste tu código a GitHub con todo lo necesario?