ISW1-2020-2C-Parcial-Enunciado.md

Adventure Games!

FooArts

FooArts es una empresa especializada en juegos de aventuras gráficas *point-and-click*, al mejor estilo **Monkey Island** y **Day of The Tentacle**.

Están comenzando a desarrollar un nuevo *engine* de aventuras gráficas basándose en el código de uno de sus juegos y nos comentan que les parece una buena oportunidad para pulir el diseño del mismo.

Hilariously Witty Engine

Nos cuentan que hasta el momento el nuevo engine llamado **HWEngine** tiene las siguientes entidades:

Backpack

Un objeto `Backpack` representa la mochila del protagonista, y sirve para guardar los elementos portables que va encontrando. Un `backpack` tiene una **cantidad máxima de peso** y una **cantidad máxima de** `puntos de juego`.

PlayableCharacter

Un objeto `PlayableCharacter` representará cualquier personaje manejado por el jugador. El personaje lleva un `backpack` y es conocido por su `name`.

StageObject

Clase raíz de la jerarquía utilizada para modelar los elementos que se encuentran en una escena. Tiene las siguientes subclases:

Door

Las puertas en una escena serán instancias de la clase `Door`. Una puerta puede estar en **3 estados** posibles:

- `Opened` (abierta)
- `Closed` (cerrada)
- `KeyClosed` (cerrada con llave)

PortableObject

Clase abstracta que representa los elementos portables, es decir, los elementos que el protagonista puede ir agregando a su `backpack`. Se divide en:

- `NoPointsPortableObject` que modela los elementos portables pero que no suman `puntos de juego`.
- `PointsPortableObject` que modela los elementos portables que suman `puntos de juego`.

Trabajo a Realizar:

1. Sacar el código repetido en los métodos de la categoría `accessing` en `Backpack`. **(3pts)**
2. Sacar el código repetido en el método de la categoría `adding` en `Backpack`. Además **la implementación resultante no debe tener** `ifs` cuando los mismos pueden ser reemplazados por polimorfismo. **(3pts)**
3. Mejorar el diseño de `Door`. **Ayuda:** realizar un diseño donde no sea necesario el uso de `ifs` en la implementación de los métodos de la categoría `playable character reactions`. Notar la existencia de una clase llamada `DoorState` que puede servir como punto de partida. **(4pts)**

Extras:

1. Sacar el código repetido de los `test07` y `test08` de `Backpack` (sin cambiar los assertions). **(1pt)**
2. Sacar el código repetido de los `test09` y `test10` de `Backpack` (sin cambiar los assertions). **(1pt)**

Usar las heurísticas de diseño vistas hasta ahora (buenos nombres, métodos cortos y categorizados, etc.)

IMPORTANTE:

1. No modificar los tests `tests01` al `test06` y del `test11` al `test15` de `Backpack`.
2. No modificar los tests de `Door`.
3. Todos los tests deben seguir funcionando.

Consultas durante el parcial:

1. Utilizar la siguiente planilla para anotarse para hacer consultas.
2. En la misma deben completar:
    i. Hora de consulta
    ii. Nombre y Apellido
    iii. Nro de Room
3. Los docentes completarán con el Docente asignado y la Hora de atención al momento de atender la consulta.

Entrega:

1. Entregar el fileout de la categoría de clase `ISW1-2020-2C-Parcial-Enunciado` que debe incluir toda la solución (modelo y tests). El archivo de fileout se debe llamar: `ISW1-2020-2C-Parcial-Enunciado.st`
2. Entregar también el archivo que se llama `CuisUniversity-nnnn.user.changes`
3. Probar que el archivo generado en 1) se cargue correctamente en una imagen "limpia" (o sea, sin la solución que crearon) y que todo funcione correctamente. Esto es fundamental para que no haya problemas de que falten clases/métodos/objetos en la entrega.
4. Realizar la entrega enviando mail a la lista de Docentes: ingsoft1-doc@dc.uba.ar con el **Subject:** LU nnn/aa - Solucion 1er parcial 2c2020

IMPORTANTE: Al enviar la solución del parcial deben recibir una confirmación de recepción.