Repetitive work

Explicación del Ejercicio 02: Trabajo Repetitivo

Este ejercicio consiste en ampliar la jerarquía de clases que has estado construyendo en ejercicios anteriores. Ahora, debes implementar una nueva clase llamada FragTrap, que **hereda** de ClapTrap.

📌 *Objetivo Principal*

Crear la clase FragTrap con las siguientes características:

- **Heredar** correctamente de ClapTrap.
- **Modificar los mensajes de construcción y destrucción** para diferenciarlos de los de ScavTrap.
- **Asegurar una correcta secuencia de construcción y destrucción** en las pruebas.
  - Cuando se crea un FragTrap, primero se construye un ClapTrap.
  - Cuando se destruye un FragTrap, primero se destruye su contenido antes que ClapTrap.
- **Atributos con valores específicos**:
  - name → Se pasa como parámetro al constructor.
  - hitPoints = 100
  - energyPoints = 100
  - attackDamage = 30
- **Añadir una nueva función miembro especial** llamada highFivesGuys(), que debe imprimir un mensaje positivo solicitando un "choca esos cinco".
- **Realizar pruebas exhaustivas** del comportamiento de FragTrap.

◇ Enfoque en C++98

1️⃣ *Herencia Correcta*

Debes definir FragTrap como una clase derivada de ClapTrap. Para ello, usa **herencia pública**:

class FragTrap : public ClapTrap {

Esto garantiza que FragTrap hereda **públicamente** todas las propiedades y métodos de ClapTrap.

2️⃣ *Constructor y Destructor*

- El constructor debe llamar explícitamente al constructor de ClapTrap.
- El destructor debe mostrar un mensaje distinto al de ScavTrap.

☑ Resumen de la Estrategia

1. **Definir FragTrap** heredando de ClapTrap.
2. **Configurar los atributos** en el constructor.
3. **Llamar al constructor de ClapTrap** en la inicialización.
4. **Definir un mensaje de destrucción distinto** para FragTrap.
5. **Implementar highFivesGuys()** para imprimir un mensaje en consola.
6. **Hacer pruebas** para verificar que el orden de construcción y destrucción es correcto.

📌 Explicación del Código

El código implementa una jerarquía de clases basada en ClapTrap, que representa un sistema de combate en el que los personajes pueden atacar, recibir daño y curarse.

Las clases principales son:

- **ClapTrap** → Clase base con atributos de combate (vida, energía y ataque).
- **ScavTrap** → Hereda de ClapTrap y agrega guardGate().
- **FragTrap** → Hereda de ClapTrap y agrega highFivesGuys().

1️⃣ *ClapTrap: La Clase Base*

Esta clase define las propiedades y métodos esenciales para un personaje de combate.

Atributos

- _name: Nombre del personaje.
- _hitPoints: Vida del personaje.
- _energyPoints: Energía disponible.
- _attackDamage: Daño que puede causar.

Métodos Importantes

- **Constructores y Destructor:**
  - **Por defecto:** Inicializa valores predeterminados.
  - **Copia:** Clona un ClapTrap existente.
  - **Con parámetros:** Recibe un nombre y lo asigna.
  - **Destructor:** Muestra un mensaje cuando el objeto se destruye.
- **Métodos de acción:**

  - attack(): Si tiene energía y vida, ataca y reduce su energía.
  - takeDamage(): Reduce la vida cuando recibe daño.
  - beRepaired(): Restaura la vida del ClapTrap.
- **Getters y Setters:** Permiten acceder y modificar los atributos de forma segura.

2️⃣ *ScavTrap: Hereda de ClapTrap*

- **Atributos heredados de ClapTrap.**
- **Métodos adicionales:**
  - guardGate(): Activa el "Modo Guardián".
- **Diferencias con ClapTrap:**
  - Modifica el comportamiento de attack() para gastar 5 puntos de energía por ataque.
  - Añade un mensaje adicional en guardGate().

3️⃣ *FragTrap: Hereda de ClapTrap*

- **Atributos heredados de ClapTrap.**
- **Método adicional:**
  - highFivesGuys(): Muestra un mensaje solicitando un "choca esos cinco".
- **Errores en la implementación:**
  - **Error en FragTrap.cpp** → Está incluyendo ScavTrap.hpp en lugar de FragTrap.hpp.
  - **Falta de sobreescritura de attack()** → No gasta más energía como ScavTrap.

4️⃣ *main.cpp: Pruebas del Código*

1. **Crea dos FragTrap:**
   a. "Damien" y "The child".
2. **Modifica los atributos:**
   a. Asigna 666 puntos de energía a "Damien".
   b. Copia los valores de tmp en "Damien".
3. **Prueba los métodos:**
   a. "Damien" ataca varios objetivos.
   b. Recibe daño y se cura.
   c. Usa highFivesGuys().