BraiiiiiiinnnzzzZ

Este ejercicio consiste en crear una clase Zombie en C++, que simula el comportamiento de los zombis con ciertas características y funcionalidades. A continuación, se explica paso a paso lo qué debes hacer:

1. Crear la clase Zombie:

   - Debes crear una clase llamada Zombie.
   - La clase debe tener un atributo privado, que será una cadena de texto (string) llamada name. Este atributo almacenará el nombre del zombi.

2. Añadir una función miembro announce():

   - Annoounce: anunciar, notificar, avisar, informar...
   - La clase debe tener una función miembro llamada announce(), que será responsable de hacer que el zombi se presente.
   - El zombi debe anunciarse diciendo su nombre seguido del sonido característico de los zombis, que es: "BraiiiiiiinnnzzzZ...".
   - El formato del mensaje es: <name>: BraiiiiiiinnnzzzZ.... No se deben imprimir los signos de menor y mayor (< y >).

   Ejemplo: En el caso de que el zombi en cuestión se llama "Foo", el mensaje que se debe mostrar será: Foo: BraiiiiiiinnnzzzZ....

3. Implementar la función newZombie():

   - Debes crear una función llamada newZombie() que reciba un parámetro name de tipo std::string.
   - Esta función debe crear un nuevo objeto de tipo Zombie, asignarle el nombre proporcionado y devolver un puntero a este objeto.
   - El propósito de esta función es crear un zombi, darle un nombre, y permitir que el puntero al zombi sea utilizado fuera del ámbito de la función.

4. Implementar la función randomChump():

   - Debes crear una función llamada randomChump() que también reciba un parámetro name de tipo std::string.
   - Esta función debe crear un zombi, asignarle el nombre y hacer que el zombi se anuncie a sí mismo usando la función announce().
   - Sin embargo, el zombi creado dentro de randomChump() debe ser destruido de manera automática cuando termine la función (es decir, no es necesario devolver un puntero, ya que el zombi se crea y destruye en el mismo ámbito).

5. Manejo de memoria: Stack vs Heap:

- El objetivo de este proyecto es que determines cuándo es mejor usar la pila (stack) y cuándo es mejor usar el montón (heap) para asignar memoria para los zombis.
- Pila (stack): Es más rápida y la memoria se libera automáticamente cuando la función termina, pero no puedes usar objetos de larga duración que necesiten ser utilizados fuera de la función que los creó.
- Montón (heap): Permite crear objetos cuyo ciclo de vida puede ser controlado manualmente. Estos objetos deben ser liberados (destruidos) cuando ya no los necesites.
- Tendrás que decidir si es mejor crear los zombis en el stack (para objetos temporales) o en el heap (cuando necesites que los objetos existan fuera del alcance de la función).

6. Destructor de la clase Zombie:

- La clase Zombie debe tener un destructor. El propósito del destructor es destruir el objeto cuando ya no se necesita.
- Además, el destructor debe imprimir un mensaje con el nombre del zombi para fines de depuración. Esto es útil para verificar que los zombis se están eliminando correctamente.

★ Resumen de lo que debes hacer:

- ✓ Crear una clase Zombie con un atributo privado name.
- ✓ Añadir una función announce() para que el zombi se anuncie con el formato <name>: BraiiiiiiinnnzzzZ....
- ✓ Implementar una función newZombie() que cree un zombi y devuelva su puntero.
- ✓ Implementar una función randomChump() que cree un zombi, lo nombre y lo haga anunciarse.
- ✓ Decidir cuándo asignar zombis en la pila o en el montón.
- ✓ Añadir un destructor que imprima un mensaje con el nombre del zombi al ser destruido.

Este proyecto tiene como objetivo familiarizarte con el manejo de punteros, la gestión de memoria dinámica (heap) y la gestión de memoria automática (stack) en C++, así como entender cómo funciona la creación y destrucción de objetos.

Zombie.hpp