PISTAS PARA EL PROYECTO

Asignatura: 750015C – FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS
Profesor: Ángel E. García Baños
Email: angel.garcia@correounivalle.edu.co
EISC – Universidad del Valle
2021-10-25

Juego: LA BARCA

PISTAS

Lo siguiente son sugerencias que facilitan llegar a la solución. Pero no es obligatorio realizarlo así.

Diseñe las clases:

- Jugador
- Orilla
- Barca
- Individuo
- Lugar

Diseñe estas relaciones:

- **Orilla** es un **Lugar**.
- **Barca** es un **Lugar**
- Debe haber 2 objetos de clase **Orilla** y un objeto de clase **Barca**.
- La clase **Lugar** conoce **Individuos** (para ello debe tener en sus atributos un vector de punteros a Individuo).
- Los demás objetos (robot, zorro, conejo, lechuga) deben ser de la clase **Individuo**.
- Cada **Individuo** conoce los **Individuos** que se puede comer (para ello, la clase Individuo debe tener en sus atributos un vector de punteros a Individuo que puede comer; o sea, el zorro puede comer conejo; y el conejo puede comer lechuga).

Diseñe estas responsabilidades:

- La clase **Lugar** es la que debe determinar si se perdió, bien sea porque allí hay un individuo que puede comerse a otro, a la vez que no está el robot, bien sea porque un Individuo saltó hacia un lugar vecino, pero el vecino es **nullptr** (o sea, se cayó al agua).

Main:

- En el programa principal debe crear los objetos de cada clase (robot, zorro, conejo, lechuga, orillaIzquierda, orillaDerecha, barca y jugador), generar las relaciones entre ellos (el zorro se puede comer al conejo etc; la barca está al lado de la orillaIzquierda, la barca está lejos de la orillaDerecha, la orillaIzquierda está al lado de la barca, etc; el jugador conoce a todos los demás) y dar la orden inicial para que comience el juego (por ejemplo, jugador.jugar() o similar).

Polimorfismo:

- Para lograr el polimorfismo se sugiere que cada **Lugar** tenga dos punteros apuntando a los otros **Lugar**es. Por ejemplo, inicialmente la **orillaIzquierda** debe tener como vecino inmediato a la **barca** y como vecino alternativo **nullptr** (el agua). La **barca** debe tener como vecino inmediato a la **orillaIzquierda**, y como vecino alternativo a la **orillaDerecha**. Y la **orillaDerecha** debe tener como vecino inmediato un **nullptr** (el agua) y como vecino alternativo la **barca**. Cada vez que la **barca** se mueva, se intercambia el vecino inmediato con el alternativo en los tres objetos.
- El polimorfismo se puede lograr entonces en la función **Jugador::jugar()** teniendo en la parte privada de **Jugador** un **vector<Lugar *>** de 3 posiciones, apuntando respectivamente a la **orillaIzquierda**, **barca** y **orillaDerecha**. Sin polimorfismo esta función saldrá muy larga y con muchos condicionales.
- Otra técnica para simplificar código es que cada Individuo contenga un string con la orden que reconoce (“R” para el robot, etc.). De este modo, se eliminan muchos if del Jugador, pues solo tiene que enviar la orden a todos los Individuos, y que ejecute la orden el que la reconozca como propia.

Extensibilidad:

- Asegúrese de que cada clase conoce todo lo suyo, es decir, sus datos y los algoritmos que manipulan esos datos. En particular, cada Individuo debe conocer su nombre, la orden que sabe ejecutar y los individuos que se puede comer.