Práctica 1. Algoritmos devoradores

Alejandro Guitarte Fernández
alejandro.guifer@alum.uca.es
Teléfono: 601048359
NIF: 32092444S

31 de octubre de 2022

1. Describa a continuación la función diseñada para otorgar un determinado valor a cada una de las celdas del terreno de batalla para el caso del centro de extracción de minerales.

   En mi caso, la función asigna valores en función de la distancia al obstáculo más cercano. De esta manera, el centro de extracción quedará ubicado junto a un obstáculo, por lo que necesitaremos menos defensas para "rodear" la mina.

   Los parámetros que recibe son:
   - Fila y columna de la celda a valorar.
   - Tamaño de las celdas, para valorar las distancias.s
   - Lista de obstáculos.

2. Diseñe una función de factibilidad explicita y descríbala a continuación.

   La función de factibilidad recibe la defensa a colocar más la celda y columna de la casilla a comprobar; así como las dimensiones del mapa, y listas de obstáculos y defensas.

   Así comprueba en primer lugar que la casilla esté dentro del mapa, sin salirse por los bordes, y luego que al colocar la defensa en esa casilla, no se salga del mapa por ser demasiado grande. Por último, recorre las listas de obstáculos y defensas y comprueba que no se produzcan colisiones en caso de insertar la defensa en la cela indicada.

3. A partir de las funciones definidas en los ejercicios anteriores diseñe un algoritmo voraz que resuelva el problema para el caso del centro de extracción de minerales. Incluya a continuación el código fuente relevante.

```
// sustituya este codigo por su respuesta
void placeDefenses(...) {

    List<Defense*>::iterator currentDefense = defenses.begin();
    while(currentDefense != defenses.end() && maxAttemps > 0) {

        (*currentDefense)->position.x = ((int)(_RAND2(nCellsWidth))) * cellWidth + cellWidth
            * 0.5f;
        ...
        ++currentDefense;
    }
}
```

4. Comente las características que lo identifican como perteneciente al esquema de los algoritmos voraces.

   Escriba aquí su respuesta al ejercicio 4.

5. Describa a continuación la función diseñada para otorgar un determinado valor a cada una de las celdas del terreno de batalla para el caso del resto de defensas. Suponga que el valor otorgado a una celda no puede verse afectado por la colocación de una de estas defensas en el campo de batalla. Dicho de otra forma, no es posible

modificar el valor otorgado a una celda una vez que se haya colocado una de estas defensas. Evidentemente, el valor de una celda sí que puede verse afectado por la ubicación del centro de extracción de minerales.

Escriba aquí su respuesta al ejercicio 5.

6. A partir de las funciones definidas en los ejercicios anteriores diseñe un algoritmo voraz que resuelva el problema global. Este algoritmo puede estar formado por uno o dos algoritmos voraces independientes, ejecutados uno a continuación del otro. Incluya a continuación el código fuente relevante que no haya incluido ya como respuesta al ejercicio 3.

```
// sustituya este codigo por su respuesta
void placeDefenses(...) {

    List<Defense*>::iterator currentDefense = defenses.begin();
    while(currentDefense != defenses.end() && maxAttemps > 0) {

        (*currentDefense)->position.x = ((int)(_RAND2(nCellsWidth))) * cellWidth + cellWidth
            * 0.5f;
        ...
        ++currentDefense;
    }
}
```

Todo el material incluido en esta memoria y en los ficheros asociados es de mi autoría o ha sido facilitado por los profesores de la asignatura. Haciendo entrega de este documento confirmo que he leído la normativa de la asignatura, incluido el punto que respecta al uso de material no original.