**INFORME DE INGENIERÍA**

**PROYECTO FINAL**

**MARIA CAMILA LENIS RESTREPO**

**JUAN SEBASTIAN PALMA GARCÍA**

**JAVIER ANDRÉS TORRES REYES**

**ALGORITMOS Y ESTRUCTURAS DE DATOS**

**2018-2**

**INFORME DE INGENIERÍA**

**Paso 1: Identificación del problema**

**Definición del problema**

Se busca implementar un mapa útil junto a un sistema de información mediante grafos para la mansión Winchester.

**Justificación**

La mansión Winchester ha existido por una gran cantidad de tiempo y al convertirse en un museo se ha decidido implementar una renovación al tipo de planificación utilizado la mansión. Primero, se ha buscado implementar un nuevo mapa que innove y facilite la movilidad de los visitantes de una sala a otra y que les sirva como guía para llegar a una sala que ellos quieran. Además, han decidido modificar el sistema de inventario de la mansión ya que se han descubierto las misteriosas desapariciones de objetos valiosos de algunas habitaciones y así evitar la perdida de estos valiosos objetos. Inclusive, se ha tomado en cuenta que la mansión al tener una gran antigüedad se requiere la demolición y reconstrucción o adición de nuevas habitaciones para que el museo logre mantenerse abierto al publico y no imponer algún tipo de riesgo a cualquier visitante. Por último, entre estos nuevos planes se ha hecho una inversión mayor en los sistemas de comunicación dentro de la mansión para permitir una mayor facilidad de comunicación entre el museo y los visitantes en caso del cierre del museo y así evitar que algunas personas terminan encerradas dentro de la casa embrujada como ha ocurrido antes.

**Requerimientos funcionales**

1. Dada una habitación de la mansión se debe encontrar el camino más rápido, en minutos, desde esa habitación hasta la salida. Si la habitación no tiene salida, de debe mostrar un mensaje de advertencia.
2. El sistema debe encontrar el camino que pase por menos habitaciones desde un punto a otro de la mansión. El usuario debe ingresar el punto de partida y el de llegada, y recibe una secuencia de habitaciones incluyendo el punto de partida y el de llegada.
3. El sistema debe transmitir el mensaje de cierre a todos los rincones de la casa, de manera que este llegue de la manera más rápida posible teniendo en cuenta lo que se demora cruzar de una habitación a otra, desde la entrada de la mansión.
4. Añadir una habitación a la mansión. La nueva habitación debe contener el indicador, las habitaciones a las cuales se puede llegar a través de ella, y las habitaciones de las cuales se puede llegar a ella.
5. Dado el indicador de la habitación se debe eliminar la habitación del mapa. Si la habitación contenía tesoros, estos deben quedar en el registro de tesoros encontrados.
6. Dado el indicador de la habitación se deben registrar tesoros encontrados. Se debe añadir el nombre y el valor del tesoro y la habitación a la cual pertenece. Si la habitación es eliminada el tesoro quedará solo en el registro y será enviado al museo.
7. Visualizar los tesoros encontrados, ya sea que aún pertenezcan a la habitación o que pertenezcan al museo. Se debe mostrar su nombre, valor, habitación a la que pertenece o, en su defecto, que pertenece al museo.

**Paso 2: Recopilación de la información**

**Grafo**

Un grafo es una estructura de datos no linear que se compone de Vértices y Aristas. Los vértices o nodos son los componentes del grafo que contienen un objeto y las aristas son caminos que conectan una vértice a otra. Un grafo puede ser direccionado o no direccionado, en el cual si este es direccionado solo puede ir de un vértice inicial a un final sin poder devolverse a la inicial a menos de que exista una arista que conecte este nodo con el inicial. El grafo no direccionado tiene aristas que permiten el viaje entre vértices mediante el uso de una sola arista ya que no tiene ningún limitante. Un grafo se considera conexo cuando se encuentra completamente conectado y no existe mas de un grafo, el no conexo esta separado en dos subgrafos.

Tomado de <https://www.geeksforgeeks.org/graph-and-its-representations/> & <https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/>

Como representar un Grafo:

* Lista de Adyacencia: Es una Lista de Listas que contiene todos los vértices y marca dentro de cada una en base a un vértice si tiene aristas que los conectan entre si.
* Matriz de Adyacencia: Es una matriz de vértices que compara cada vértice con otro y si alguno tiene un vértice que lo conecta con otro se marca dentro de la intersección de la matriz la existencia de la arista, con su peso o un uno.

**DFS**

DFS o Depth First Search, es un algoritmo que crea un recorrido desde un nodo inicial A, hacia el resto sin repetir los nodos que visita. Este algoritmo recorre la profundidad de un nodo hasta que no existan más nodos por los cuales se puede recorrer, antes de pasar al siguiente. Este método utiliza un arreglo booleano que mantiene un chequeo de los nodos o vértices ya visitados por la búsqueda. Al final se retorna un árbol con los órdenes de los vértices mediante profundidad.

Tomado de <https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/>

**BFS**

BFS o Breath First Search, es un algoritmo que recorre desde un nodo inicial A, a todos los demás, sin repetir visitas. Este algoritmo recorre todos los nodos adyacentes antes de seguir bajando niveles dentro de cada nodo. Este método utiliza un arreglo booleano que mantiene un chequeo de los nodos o vértices ya visitados por la búsqueda. Al final se retorna un árbol con los órdenes de los vértices mediante su nivel de adyacencia.

Tomado de <https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/>

**Recorrido de Camino Mínimo**

Existen tres algoritmos de búsqueda de camino mínimo:

* Disjtrak: Este es un algoritmo que genera un árbol de camino más corto en el cual empieza desde un Vértice Inicial y se asignan todos los valores a cada vértice como infinito o un numero extremadamente grande, y se empieza a recorrer cada vértice sumando sus pesos y asignándoselos al mismo, y evalúa si otro nodo por otro lado llega a ese mismo vértice y evalúa cual es menor. Al final arma un árbol con los caminos más bajos desde el nodo elegido.

Tomado de <https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/>

* Bellman-Ford: Este es un algoritmo que logra hacer lo mismo que el algoritmo de Disjtrak, solo que este es capaz de manejar los vértices que tienen pesos negativos sin llegar a un bucle infinito. Este método consiste en crear una lista de todos los vértices que contendrá la distancias de los mismos a la inicial. Todas las distancias se colocan como infinitas excepto la de la inicial y luego se van llenando las distancias con un recorrido, y al mismo tiempo se evalúa si por otro vértice existe una conexión y este camino tiene menor peso al ya establecido
  + Este método maneja los pesos negativos a través de una comparación en la cual la distancia de V es mayor a la distancia de U mas la arista entre UV, existe un valor negativo.

Tomado de <https://www.geeksforgeeks.org/bellman-ford-algorithm-dp-23/>

* Floyd-Warshall: este método genera un bosque de arboles de caminos más cortos, y lo aplica para cada uno de los vértices. Este algoritmo mediante una matriz de vértices para buscar el camino más corto en cada árbol y así crear el bosque de arboles. Utiliza un método muy parecido al de Disjtrak para evaluar las distancias dentro de los arboles.

Tomado de <https://www.geeksforgeeks.org/floyd-warshall-algorithm-dp-16/>

**Arboles de Recubrimiento**

**Paso 3: Búsqueda de soluciones creativas**

**Paso 4: Transición de ideas a los diseños preliminares**

**Paso 5: Evaluación o selección de la mejor solución (Criterios y selección)**

Criterio A:

* [3] Valoración

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Criterio A | Criterio B | Criterio C | Criterio D | Total |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Paso 6: Preparación de informes**

**Diagrama de clases de la solución**

**Diagrama de objetos**

**Diseño de casos de pruebas unitarias**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba 1: | | | | |
| Clase | Método | Escenario | Entrada | Resultado |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Bibliografía**