Trabajo Final de Estructuras de Datos
Año 2021

Escape House es un juego que consiste en encontrar una estrategia para escapar de una casona mediante la resolución de desafíos o acertijos, que permiten pasar de una habitación origen a otra con puerta de salida al exterior de la casa, pasando por habitaciones intermedias y acumulando determinado puntaje. Cada habitación está comunicada con otras mediante puertas que se pueden abrir solo si se resuelven desafíos asociados. Cada acertijo resuelto otorga un puntaje que el equipo participante va acumulando y que permite pasar a alguna de las habitaciones contiguas hasta que el puntaje sumado supere el exigido y se pueda salir de la casa y ganar el juego. Cada desafío puede resolverse una única vez, por lo tanto, una vez resuelto, quedará marcado como tal por el resto del juego y el puntaje se acumulará solo una vez.

Se cuenta con un plano de la casa con las diferentes habitaciones, las puertas de cada una que comunican con las demás y su puntaje.

Para la implementación del Escape House se deberán tener en cuenta las siguientes observaciones:

- Se debe usar un grafo etiquetado que almacenará el plano de la casa con la conexión entre las distintas habitaciones. Los arcos llevarán el puntaje exigido para pasar de una habitación a otra. El sentido de circulación entre las habitaciones es indistinto.
- La información de cada habitación (código, nombre, planta, metros cuadrados, y si tiene o no salida al exterior de la casa) se debe guardar en una Tabla de búsqueda implementada con un AVL.
- Además se utilizará una tabla de búsqueda implementada con AVL para almacenar los desafíos: puntaje que otorga, nombre, tipo (lógico, matemático, búsqueda, destreza, etc). No puede haber desafíos con puntajes iguales.
- En otra tabla de búsqueda implementada con Hash(*) se almacenarán los equipos que participan: nombre del equipo, puntaje exigido para salir de la casa, puntaje total (acumulado en lo que va del juego), habitación en la que se encuentran actualmente y puntaje actual (acumulado dentro de la habitación en donde están ubicados).
- En un mapeo 1 a muchos guardar los desafíos que resolvió cada equipo implementado con Hash(*)

  (*) Puede implementarlo con hash abierto o usar la clase HashMap de Java

Se pide el desarrollo de este juego con un menú de opciones que permitan realizar las siguientes tareas:

1. Carga inicial del sistema: se debe ingresar a partir de un archivo de texto

2. Altas, Bajas y Modificaciones (ABM) de habitaciones, desafíos y equipos: En modificaciones no debe permitir modificar los atributos clave

3. Consulta sobre habitaciones:

- ***mostrarHabitación***: Dado un código de habitación, mostrar toda la información sobre la misma
- ***habitacionesContiguas***: Dado un código de habitación, mostrar las habitaciones contiguas a las que se puede acceder, y qué puntaje se necesitaría para pasar a cada una
- ***esPosibleLlegar***: Dados los códigos de ***hab1*** y ***hab2***, y un valor ***k***, mostrar si es o no posible llegar de ***hab1*** a ***hab2***, acumulando ***k*** puntos
- ***maximoPuntaje***: Dados dos códigos de habitación, mostrar el máximo puntaje que deberían acumular para ir de ***hab1*** a ***hab2 (**)***
- ***sinPasarPor***: Dados tres códigos de habitación y un valor numérico ***P***, mostrar todas las formas de llegar desde ***hab1*** a ***hab2*** sin pasar por la tercera habitación (***hab3***) que no requieran ganar más de ***P*** puntos.

4. Consultas sobre desafíos:

- ***mostrarDesafío***: Dado un código de desafío, mostrar toda su información.
- ***mostrarDesafíosResueltos***: Dado un equipo ***eq***, mostrar todos los desafíos que ya resolvieron
- ***verificarDesafíoResuelto***: Dado un equipo y un nombre de desafío, indicar si el equipo ya lo resolvió ***(**)***
- ***mostrarDesafíosTipo***: Dados dos puntajes ***a*** y ***b*** y un tipo ***X***, mostrar todos los desafíos de tipo X con puntaje en el rango ***[a, b]*** (por ejemplo, listar todos los desafíos de tipo ***lógico*** con puntaje entre 30 y 55)

5. Consultas sobre equipos participantes:

- ***mostrarInfoEquipo***: Dado el nombre del equipo, mostrar todos sus datos.
- ***posiblesDesafios***: Dado un equipo y una habitación ***hab***, en caso en que ***hab*** sea adyacente al lugar donde esté ubicado el equipo, mostrar todos los desafíos que podría resolver el equipo para pasar a ***hab*** resolviendo un solo desafío. En caso en que ***hab*** no sea adyacente, mostrar un mensaje aclaratorio ***(**)***
- ***jugarDesafío*****:** Dado un equipo ***eq*** y un desafío, marcar el desafío como ganado y actualizar los datos del equipo apropiadamente.
- ***pasarAHabitacion*****:** Dado un equipo ***eq*** y una habitación ***hab***, verificar si es posible que el equipo ***eq*** pase a la habitación ***hab*** (considerando si es contigua a la actual y el puntaje acumulado es suficiente) y en caso afirmativo actualizar los datos del equipo apropiadamente.
- ***puedeSalir***: Dado el nombre del equipo participante, decir si puede o no salir del juego en base al puntaje acumulado, al puntaje que debe obtener para ganar el juego y si la habitación en la que se encuentra tiene o no salida al exterior.

6. Consulta general:

- ***mostrarSistema***: Operación que permite ver todas las estructuras utilizadas con su contenido para verificar que se encuentran bien cargadas (toString de cada estructura donde sea visible la ubicación de los nodos en el árbol AVL, orden de vértices y sus adyacentes en el grafo, y pares asociados en los mapeos.

Requisitos del programa

- El programa debe permitir la ejecución por separado de cada una de las operaciones especificadas.
- El programa debe ser eficiente: Debe recorrer las estructuras sólo lo necesario y haciendo buen uso de la memoria.
- Las estructuras deben estar implementadas de forma genérica para elementos de tipo Object o Comparable de Java, según la necesidad de la estructura.
- **La carga inicial del sistema** debe realizarse a partir de un archivo de texto con formato preestablecido.
- **Utilizar un archivo de log (archivo de texto)** para guardar la siguiente información: estado del sistema al momento de terminar la carga inicial, anotar qué operaciones de ABM se realizan a lo largo de la ejecución (Ej: “Se crea la habitación 01”, “Se borró el desafío D1”, etc), y el estado del sistema completo al momento de terminar de ejecutarse.

Condiciones y fechas de entrega

- El programa debe realizarse de manera individual y además de subirlo a PEDCO debe presentarse a los docentes personalmente.
- Al momento de la defensa, deberá presentar en papel el dibujo de la casa cargado (grafo) y la información de las puertas (AVL)
- Los **estudiantes que promocionan** la materia tendrán tiempo de entrega hasta el 6 de agosto de 2021. No necesitan realizar los módulos marcados con ***(**)***
- Los estudiantes que no promocionan podrán entregarlo en cualquier momento, pero como mínimo deberán hacerlo 2 semanas antes de presentarse a rendir el final regular.

PARA INVESTIGAR

- Leer sobre StringTokenizer (fraccionar un String con un caracter separador)
- Abrir archivo de texto: FileReader, BufferedReader
- Escribir archivo de texto: FileWritter

FORMATO DE ARCHIVO DE TEXTO (EJEMPLO)

Habitación: código, nombre, planta, metros cuadrados, tiene salida al exterior

- H;1;Comedor;0;20; false
- H;2;Cocina;0;15;true
- H;5;Sótano;-1;30;false

Equipo: nombre del equipo (único), puntaje exigido para salir de la casa, puntaje total (acumulado en lo que va del juego), habitación en la que se encuentran actualmente y puntaje actual (acumulado dentro de la habitación que están ubicados).

- E;Los Valientes;400;130;4;0
- E;Los Locos Adams;400;50;2;10

Desafío: puntaje que otorga, nombre y tipo (lógico, matemático, letras, búsqueda, destreza, ingenio, etc)

- D;30;Acertijo;Lógico

- D;50;Sudoku;Matemático
- D;65;Trabado;Ingenio

PUERTAS (ARCOS DEL GRAFO)

- P:1;2;40
- P:2;3;60

INSERTAR EN LAS ESTRUCTURAS **AL MENOS** 20 ELEMENTOS DE TIPO DESAFIO, 20 HABITACIONES Y 10 EQUIPOS.

ASEGURAR QUE EN EL SET DE CARGA INICIAL LOS ELEMENTOS SE LISTEN EN FORMA DESORDENADA PARA QUE SE PRODUZCAN TODAS LAS ROTACIONES POSIBLES EN CADA AVL.

Listado de habitaciones o lugares posibles de la casona

1. Altillo
2. Baño
3. Biblioteca
4. Bodega
5. Cochera
6. Cocina
7. Comedor
8. Depósito
9. Despensa
10. Dormitorio principal
11. Dormitorio de huéspedes
12. Escalera
13. Escritorio
14. Gimnasio
15. Invernadero
16. Lavadero
17. Natatorio
18. Pasillo
19. Quincho
20. Recibidor
21. Sala de Estar
22. Sala de Juegos
23. Sala de Música
24. Sótano