**Instituto Costarricense de Costa Rica**

**Unidad de Computación**

**Proyecto N°2: Laberinto Grafos**

**(“Maze Runner”)**

**José Daniel Amador Salas**

**José Pablo Brenes Alfaro**

**Sede San Carlos**

**15/11/2017**

**Contenido**

[Introducción 3](#_Toc498542411)

[Análisis del Problema 4](#_Toc498542412)

[Solución del problema 5](#_Toc498542413)

[Análisis de los resultados 7](#_Toc498542414)

[Conclusiones 9](#_Toc498542415)

[Bibliografía 10](#_Toc498542416)

# Introducción

Con el objetivo de que los estudiantes conozcan sobre el manejo de estructuras de datos. Se propuso el desarrollo de un proyecto que involucre el manejo de grafos (“...conjunto, no vacío, de objetos llamados vértices (o nodos) y una selección de pares de vértices, llamados aristas...” (Jacv193.blogspot.com. 2011) y árboles (“estructuras de datos que se adapta a la representación natural de informaciones homogéneas organizadas” (Garo. 2011). El proyecto consiste en un programa creado en el lenguaje de programación Java, que tiene la temática de un laberinto, donde varios personajes tendrán que encontrar el camino más corto para llegar a un objetivo dentro del mapa. En su recorrido se encontrar con diferentes obstáculos (paredes y tiempos de espera) y algunos bonus especiales que le ayudarán a llegar más rápido al objetivo.

El personaje tendrá que comparar las rutas y encontrar la que menor peso tenga para así llegar sin demoras al objetivo, también contará con una función de teletransporte que le permitirá al personaje adelantarse a los demás jugadores, pero si este hace que el personaje se aleje del objetivo, se debe buscar un camino mejor. Para la simulación de los caminos se utilizó una matriz gráfica donde cada elemento de la matriz es un vértice de un grafo y de esta manera se pudo realizar un algoritmo que determinarse la ruta más corta. Los personajes tendrán dentro del programa movimiento automático hacia el objetivo, este se realiza mediante la implementación de hilos de ejecución, los cuales permiten realizar tareas casi simultáneas.

# Análisis del Problema

Se requiere el desarrollo de un programa que permita la implementación de las estructuras de datos (Grafos, Árboles), para la simulación de recorridos dentro de mapas. El programa se debe desarrollar en Java y debe permitir la creación de mapas (estilo laberinto) de manera aleatoria. Para que cumpla los requerimientos necesarios debe incluir algunos aspectos como: debe permitir la existencia de bonus aleatorios en algunas de las rutas posibles, la pared del laberinto también se debe de generar aleatoriamente, así como la cantidad de personajes que aparecerán en el mapa.

Entre los procesos de la aplicación también está almacenar información de los bonus del programa estructurados como un árbol de datos, esta información debe persistir aun después de que el programa termina de ejecutarse, pueden existir diferentes tipos de bonus dentro del mapa, estos deberán realizar diferentes acciones sobre el personaje o los otros. El objetivo principal será que cada personaje deberá llegar a un objetivo generado aleatoriamente en el mapa, buscando siempre la ruta más corta hacia el objetivo (por saltos o peso), si aparecen bonus en el camino, determinar si estos son buenos o malos para el personaje en su recorrido. Los personajes que estén dentro del mapa deben moverse de manera simultánea hasta llegar al objetivo.

# Solución del problema

Para solventar el planteamiento del proyecto, se desarrolla una aplicación en java en la cual se basa en la utilización de las estructuras de datos conocidas como grafos y árboles. Por medio de una interfaz gráfica se crea el mapa, la cual es un laberinto o caminos que están separados por paredes que estarán generados aleatoriamente (usando un algoritmo de colocación de paredes por medio las filas y columnas lógicas de cada pared), estos son vértices que estructuralmente forman un tipo de matriz en la cual los personajes se podrán mover ( por medio de caminos) obteniendo las posiciones de cada vértice, que en adición, cada vértice posee un booleano que permite conocer si es camino o no (estos pintados en un panel) para que cada personaje que sale aleatorio en el mapa (estos pueden ser dos o tres) puedan ir al destino que determina el final del recorrido de cada personaje.

Para que estos puedan recorrerlo, se necesita que cada vértice esté conectado con el siguiente y así al inverso teniendo en cuenta que estos tienen un peso en común para poder realizar su debido recorrido de la manera más corta y que estos enlaces se realizaran si el vértice es camino y no pared. También es importante saber que el método empleado para realizar la ruta corta dentro del grafico se basa no en su totalidad en el método de profundidad en un grafo que se básicamente recorre todos los caminos posibles desde un punto de partida, que modificando una lista en la cual va agregando los recorridos, agregando un destino de método de parada y un contador de peso, se realiza correctamente la búsqueda de la ruta más corta (está contenida en un arreglo que después se leerá para ejecutarla) de un punto a otro basándose en su peso y evitando los bonus que son malos para los personajes.

Estos bonus que contienen un id de identificación y un peso, se basan en habilidades que pueden ayudar al personaje a llegar aún más rápido si son buenos o aún más despacio si son malos, que en resumen son: aceleración, ralentización para los otros jugadores, cambiar el objetivo de lugar(recalculando las rutas de cada personaje) que serán los positivos que por otra parte el de esperar N segundos (que se obtienen aleatoriamente) es negativo, también están los pasivos que serían el de teletransportar el personaje que dependiendo de la distancia de este se considera malo o bueno, y por último el aleatorio que este si se pasa por él, sale de manera aleatoria cualquiera de los bonus mencionados anteriormente.

Los bonus estarán almacenados en un árbol que por medio de su id se colocan en el de manera ordenada, estos guardados en un archivo binario de manera post-orden que al iniciar la aplicación se cargaran en el árbol y se obtienen de manera al azar para colocarlos en las posiciones de los caminos que salieron del resultado dado que cuando un personaje lo tome este desaparece del mapa. Los personajes como se mencionan anteriormente salen al azar y pueden ser dos o tres dependiendo de otro resultado aleatorio, estos caminan de forma automática hasta su destino por medio del método explicado anteriormente llegan a un destino en común, basándose en la utilización de hilos para cada movimiento paralelo de cada personaje, estos como van siguiendo el recorrido nunca van a traspasar paredes, pero si los demás personajes.

# Análisis de los resultados

La siguiente tabla se basa en el análisis de cada sección que se menciona previamente en el enunciado del proyecto. Esta se divide en cuatro partes, la primera se basa en lo que vamos a evaluar que se obtuvo del enunciado, después vienen las dos secciones que indican si se completó o no la evaluación por medio de una X que selecciona el resultado y por último es la sección de observaciones que se tuvo al realizar la evaluación.

|  |  |  |
| --- | --- | --- |
| **Evaluar** | **Completado** | **Observaciones** |
| Bonus almacenados en archivo | Se realizo al 100 % | Los bonus se almacenan en un archivo binario |
| Bonus almacenados en árbol | Se realizo al 100 % | Estos están colocados dependiendo de su id |
| Se crear paredes, aleatoriamente | Se realizo al 100 % | Mediante un método que realiza un random en fila y columna |
| Objetivo random | Se realizo al 100 % | Mediante un método público que obtiene el límite y retorna el numero al azar dentro del rango |
| Cantidad de personajes random | Se realizo al 100 % | Pueden salir 2 como mínimo y 3 como máximo |
| Se establece cantidad de bonus random | Se realizo al 100 % | Los bonus salen random dentro de los diferentes caminos |
| El bonus es elegido aleatoriamente según el id | Se realizo al 100 % | Se obtiene un número del 1 al 6 y por medio de ese número se obtiene el bonus |
| Personaje mínimo 2 y máximo 3 | Se realizo al 100 % | Solo pueden haber de 2 a 3 personajes en el grafo |
| Caminar hacia el objetivo y movimiento automático | Se realizo al 100 % | Se realiza por medio de un recorrido de una lista que contiene los vértices del recorrido |
| Buscar ruta más corta | Se realizo al 100 % | Por medio de dos listas, una auxiliar y otra principal, se obtiene la ruta más corta, similar al método de profundidad |
| Caminos generados a partir de la posición del personaje | Se realizo al 100 % | Se guardan en una lista |
| Personajes se mueven simultáneos | Se realizo al 100 % | Por medio de la implementación de hilos |
| Caminos con bonus negativo, elegidos según peso | Se realizo al 100 % | La ruta más corta se basa en el peso y en el bonus negativo para realizar la ruta |
| Bonus de Aceleración | Se realizo al 100 % | Disminuye 700 el tiempo de espera |
| Bonus Teletransporte | Se realizo al 100 % | Se teletransporta el personaje dentro de dos portales |
| Bonus Espera | Se realizo al 100 % | Random entre 3000 y 4000 milisegundos |
| Bonus Ralentizar | Se realizo al 100 % | Aumenta 300 el tiempo del personaje |
| Bonus cambio de lugar | Se realizo al 100 % | Se cambian la posición donde se encuentra el destino de llegada y se recalculan las rutas |
| Bonus Random | Se realizo al 100 % | Se obtiene algún bonus por medio de un random |
| Bonus activados solo una vez | Se realizo al 100 % | Se desaparece el bonus si el personaje cae en la misma posición del bonus |

# Conclusiones

Se concluye que se pueden calcular rutas implementando la estructura de datos conocida como grafos y que perfectamente se combinan con otras estructuras como los árboles para realizar algoritmos como laberintos entre otros. La utilización de hilos es otra estructura que es muy usada en este proyecto ya que sin ella solo se podría ver el recorrido en un solo personaje y después el otro, no simultáneamente, es por ello por lo que desde hace muchos años han sido una tendencia para optimizar los procesos y darle más dinamismo a los programas que comúnmente son muy secuenciales y poco eficientes. Además, el uso de las rutas es muy usado especialmente en el ámbito de los sistemas de navegación ya que estos se basan principalmente en cómo llegar de un punto a otro en el menor tiempo y distancia posible.

En resumen, el proyecto abarco todos los términos propuestos en clases, que además de esto, se experimentó en la creación de métodos como el de las rutas cortas o el diseño de patrones tipo laberinto que ayudan a mejorar en la lógica de los programadores, así como en la innovación de los mismos.

# Bibliografía

* Jacv193.blogspot.com. (2011). *Grafos*.  Available at: <http://jacv193.blogspot.com/2011/06/grafos.html> [Accessed 31 Oct. 2017].
* Garo. (2011). *Árbol (informática)*. [online] Estructura-de-datos-garo.blogspot.com. Available at: <http://estructura-de-datos-garo.blogspot.com/2011/10/el-arbol-es-una-estructura-de-datos.html> [Accessed 31 Oct. 2017].