

Facultad De Ingeniería

Escuela De Ingeniería de Sistemas

Análisis y Diseño de Algoritmos

Mérida – Venezuela

**Análisis y Diseño de Algoritmos**

**Desarrollo de un Juego de Laberintos usando la Teoría de Grafos**

Hernández Sarait C.I: 23.722.515

Contreras Angely C.I: 23.493.654

Mejía Gustavo C.I: 25.302.093

Rojas Víctor C.I: 23.391.502

Mérida- Venezuela diciembre, 2017

**INDICE.**

**CAPITULO I**

**MARCO TEÓRICO**

|  |  |
| --- | --- |
| **1.- La Teoría de Grafos** |  |
| **1.1.- Conceptos básicos de la Teoría de Grafos** |  |
| **2.- El Laberinto.** |  |
| **3.- Videojuego** |  |
| **4.- El Diseño del Juego.** |  |
| **4.1.- Los aspectos preliminares del Video juego “MAZEOUT”** |  |
| **4.2.- Funcionamiento interno del Videojuego** |  |
| **4.3.- Lenguaje de programación del juego.** |  |
| **4.4.- La interfaz gráfica** |  |
| **4.5.- Componentes del Video juego** |  |
|  |  |
|  |  |

**CAPITULO I**

**MARCO TEÓRICO**

El capítulo que a continuación se desarrolla contiene la revisión de investigaciones previas enmarcadas en el tópico de investigación. Para su selección se ha tomado los principales debates que se están dando en la actualidad, en Teoría de Grafos, desarrollo de videojuegos, lenguaje de programación. Esta primera parte contiene los aspectos más importantes del juego.

**1.- La Teoría de Grafos.** La teoría de grafos, es una rama de las matemáticas y las ciencias de la computación que estudia las propiedades de los grafos. La teoría de grafos tiene sus fundamentos en las matemáticas discretas y las matemáticas aplicadas. Esta teoría requiere diferentes conceptos de área como combinatoria, álgebra, probabilidad, geometría de polígonos,  aritmética y topología. Actualmente ha tenido mayor influencia en el campo de la informática, las ciencias de la computación y telecomunicaciones. En la opinión de León (2011):

Entre nuestros sentidos sensoriales parece que la vista es la que ocupa la mayor parte de nuestra percepción. Nuestra memoria y recuerdo están impregnados en gran medida de imágenes visuales. En muchas ocasiones preferimos tratar con imágenes porque nos son más concretas a nuestra percepción y entendimiento que los conceptos abstractos. Cuando lidiamos con lo puramente abstracto, es muy posible que asociásemos una imagen visual. Cuando escuchamos tres por ejemplo es posible que se nos aparezca una imagen relacionada a una triplicidad o al digito tres. En la construcción del conocimiento humano suele utilizarse imágenes artificiales para expresar mejor ideas y conceptos o para sintetizarlas en una imagen que englobe y resuma los asuntos de interés. Las imágenes de este tipo son llamadas comúnmente gráficos. (p.189).

El mismo autor señala que la etimología de la palabra grafo deriva del griego y significa “escritura”. Los gráficos constituyen una de las primeras formas humanas de representar conocimiento y la computación y la programación no escapan de lo gráfico.

**1.1.- Conceptos básicos de la Teoría de Grafos**. Comenzaremos haciendo una revisión de los conceptos básicos de la Teoría de Grafos

**a.- Grafo:** es un conjunto de puntos (llamados elementos, vértices, nudos o nodos) con líneas que unen pares de vértice de ellas;  un grafo **{\displaystyle G=(V,E)}G<V, E>** es una pareja ordenada en la que **V{\displaystyle V}**es un conjunto no vacío de vértices y **{\displaystyle E}E**es un conjunto de aristas. Donde **{\displaystyle V}V**consta de pares no ordenados de vértices, tales como **{{\displaystyle {x,y}}x, y}{\displaystyle \in E}ϵ E** entonces decimos que **{\displaystyle x}x** e **{\displaystyle y}y** son adyacentes; y (en el grafo) se representa mediante una línea no orientada que une dichos vértices.

**b.- Multígrafo:** Es aquel grafo en donde dos vértices se pueden conectar por más de una arista.

**c.- Pseudografo:** Es aquel multígrafo en donde al menos exista un bucle (un vértice conectado con si mismo).

**d.- Subgrafo de un grafo:** si **G = (V, E)** es un grafo, entonces **G1 = (V1, E1)** se dice subgrafo de **G** si **V1** es un subconjunto de V, distinto del conjunto vacío y **E1** es un subconjunto de **E**, donde cada arista de **E1**.

**e.- Grafo dirigido:** Un grafo dirigido o dígrafo es un grafo **G = (V, E)** en el que **V ≠ Ф** y **E ⊆ {(a,b)∈ V ×V : a ≠ b}** es un conjunto de pares ordenados de elementos de **V**.

En un dígrafo, dada una arista **(a, b), a** es su nodo inicial y **b** su nodo final. Es decir, cada arista tiene definido un sentido. En este tipo de grafo se pueden definir dos conceptos:

**• Grado de entrada de un vértice**: es el número de aristas que llegan a dicho vértice.

• **Grado de salida de un vértice:** es el número de aristas que salen de dicho vértice.

Otros conceptos básicos de esta teoría son los siguientes:

• **Camino en un grafo**: es una sucesión finita y alternada de vértices y aristas de dicho grafo. Los extremos del camino son el vértice inicial y el vértice final. Si éstos coinciden, diremos que el camino es cerrado.

• **Grafo conexo:** es un grafo en el que dados dos vértices cualesquiera, siempre existe al menos un camino que los une.

• **Ciclo:** es un camino cerrado de un grafo donde los únicos vértices repetidos son el primero y el último.

• **Camino euleriano:** es un camino o circuito del grafo que contiene todas las aristas, apareciendo cada una de éstas exactamente una vez. Un grafo que admite dicho circuito se denomina grafo euleriano.

• **Árbol**: es todo grafo conexo y sin ciclos.

• **Grafos equivalentes**: dos grafos se dicen equivalentes si pueden obtenerse uno del otro por medio de transformaciones en las aristas, del tipo de estirarlas o hacerlas más pequeñas, pero nunca cortarlas o unirlas.

Gracias a la teoría de grafos se puede dar solución para modelar diversas estructuras como es el caso de un laberinto. A continuación se desarrolla un concepto general del tópico.

**2.- El Laberinto.**

Un laberinto es un lugar formado por calles y encrucijadas, intencionadamente complejo para confundir a quien se adentre en él. Un laberinto es un problema de topología. Para el desarrollo de este proyecto se considera uno de los principales componentes teóricos porque a partir de allí se diseña el juego. A grandes rasgos es importante revisar algunos aspectos en esta materia.

El primer atisbo de laberinto apareció muy temprano. Data de la Edad de Bronce y pueden encontrarse algunos tallados en rocas en Pontevedra, así como también en Val Camonica, en Italia. Los laberintos, con sus caminos tortuosos, han simbolizado en muchas ocasiones, un reto que nos lleve hasta una meta. De ahí, que tuviesen un significado espiritual en muchas culturas. Así, para los cristianos, el laberinto mostraba el duro camino hasta Dios. Otros han sido usados en ceremonias, rituales y danzas, o para atrapar los malos espíritus. Para los griegos, (Wills, Fernando y otros 2006), el laberinto es un palacio construido por Dédalo junto al Cnossos a petición del Rey Minos, para ser morada del minotauro. Era una red inextricable de salas y corredores de las que solamente logró salir Teseo gracias al hilo conductor que le proporcionó Ariadna.

En el caso de la investigación que se desarrolló, la primera inquietud sobre laberintos es cómo decidir qué camino tomar. Los puntos de intersección son una guía para saber dónde estuvimos. Entonces podemos marcar estos puntos y los caminos que los conectan. Adoptando esta actitud y trazando en el papel obtenemos lo que llamamos grafo; los puntos de intersección son llamados nodos y los caminos se denominan arcos.

**3.- Videojuego.**

Antes de definir el concepto es importante resaltar que la derivación de juego proviene del latín *iocus,* según Wills, Fernando y otros (2006), es un ejercicio recreativo sometido a reglas y en el cual se gana o se pierde. Existen diversidad de juegos: de azar, de ingenio, de manos, de palabras, de pelota, de rol, olímpicos, electrónicos. Estos últimos se caracterizan por ser juegos que aplican la electrónica.

**4.- El Diseño del Juego.**

**4.1.- Los aspectos preliminares del juego.**

Se consideró la temática del laberinto toda vez que es un tipo de juego que permite el desarrollo de habilidades cognitivas como planteamiento de hipótesis, análisis, síntesis, pensamiento recursivo, pensamiento estratégico.

**4.1.1 La narrativa del juego.**

Fue necesario crear un texto con personajes principales y secundarios que se circunscribieran a diversas tramas que generan en el lector variedad de emociones. La trama de MazeOut está desarrollada en torno a Psycho, el personaje principal del videojuego. Psycho, quien comienza como un niño con una vida común, se ve afectado por un hecho que marcaría el curso de su vida. Psycho, quien crecerá y se convertirá en Dr. Psycho, desarrolla un invento que revoluciona por completo el mundo, pero deberá librarse de quienes querrán eliminarlo o adueñarse de este.

Para más detalles leer en el repositorio el siguiente enlace:

https://github.com/mator0296/MazeOut

**4.1.2 El proceso de digitalización del video juego.**

A partir de la historia se definieron los personajes principales, los enemigos, los diversos escenarios, colores, tipo de letra, los videos donde se explica la historia de cada nivel.

El usuario debe conseguir el camino correcto para recorrer el laberinto (el cuál se genera aleatoriamente, cada nivel posee distintas tramas y los obstáculos a enfrentar se definen de acuerdo a la historia del juego) hasta su salida teniendo que huir de enemigos, consiguiendo recursos para la creación de la formula y otras misiones que se le presentan. El juego acaba cuando el usuario ha completado todos los niveles.

**4.2.- Funcionamiento interno del juego**

El enfoque usado para la realización del juego MazeOut, consiste en el uso de grafos conexos dirigidos. La implementación de estos grafos conexos dirigidos se ejecuta con una versión aleatoria del algoritmo de búsqueda en profundidad, usando backtracking de forma recursiva. El recorrido en profundidad es un algoritmo de búsqueda no informada utilizado para recorrer todos los nodos de un grafo o árbol de manera ordenada, pero no uniforme. El backtracking es una estrategia para encontrar soluciones a problemas que satisfacen restricciones, esta consiste en una búsqueda en profundidad.

Durante la búsqueda, si se encuentra una alternativa incorrecta, la búsqueda retrocede hasta el paso anterior y toma la siguiente alternativa. Cuando se han terminado las posibilidades, se vuelve a la elección anterior y se toma la siguiente opción. Si no hay más alternativas la búsqueda falla. De esta manera, se crea un árbol implícito, en el que cada nodo es un estado de la solución. A continuación el pseudo-código para generar el laberinto:

mazeout(graph G , carácter \*\*MAP , arc Y)

si no (arcodisponible(V))

retornar

fin si

listarcdisp = buscarc(arc , graph MAP)

repita mientras (listarcdisp != vacia)

int = rand %listarcdisp.tam

arcdest = listarcdisp.get(int)

eliminar(arcdest)

si (arcdest está disponible)

cambiomapa(arcdest , Y , MAP)

fin si

mazeout (G , MAP , arcdest)

fin repita

fin mazeout

El espacio del laberinto es considerado como un grafo cuya cantidad de nodos es n y de arcos es , por lo cual, el tamaño total del mapa será (2n+1). En informática, y más específicamente en el campo de las estructuras de datos, un nodo es un registro que contiene un dato de interés y al menos un puntero para referenciar (apuntar) a otro nodo. Un arco o arista, corresponde a una relación entre dos nodos o vértices de un grafo. Cada arco del grafo representa un espacio con cuatro paredes a su alrededor. A partir de la selección aleatoria de un arco, se agregan los arcos vecinos que no han sido visitados a una lista para luego visitarlos recursivamente, el recorrido horizontal o vertical de un arco a otro, define las paredes a quitar; una arista que no tiene vecinos no visitados se considera como un callejón sin salida, este proceso continua hasta que todos los arcos han sido visitados.

La recursividad es una técnica de programación que se utiliza para realizar una llamada a una función desde ella misma, de allí su nombre. Resolver un problema mediante recursividad significa que la solución depende de las soluciones de pequeñas instancias del mismo problema.

**4.3.- Lenguaje de programación del juego.**

El juego se desarrolla en el lenguaje de programación C++ con la implementación de la biblioteca Aleph-w. C++ es un lenguaje de programación creado con la intención de implementar al lenguaje de programación C mecanismos que permitan la manipulación de objetos. C++ es el lenguaje más usado en el desarrollo de videojuegos, pues combina el bajo nivel y la eficiencia del lenguaje C con la programación orientada a objetos.

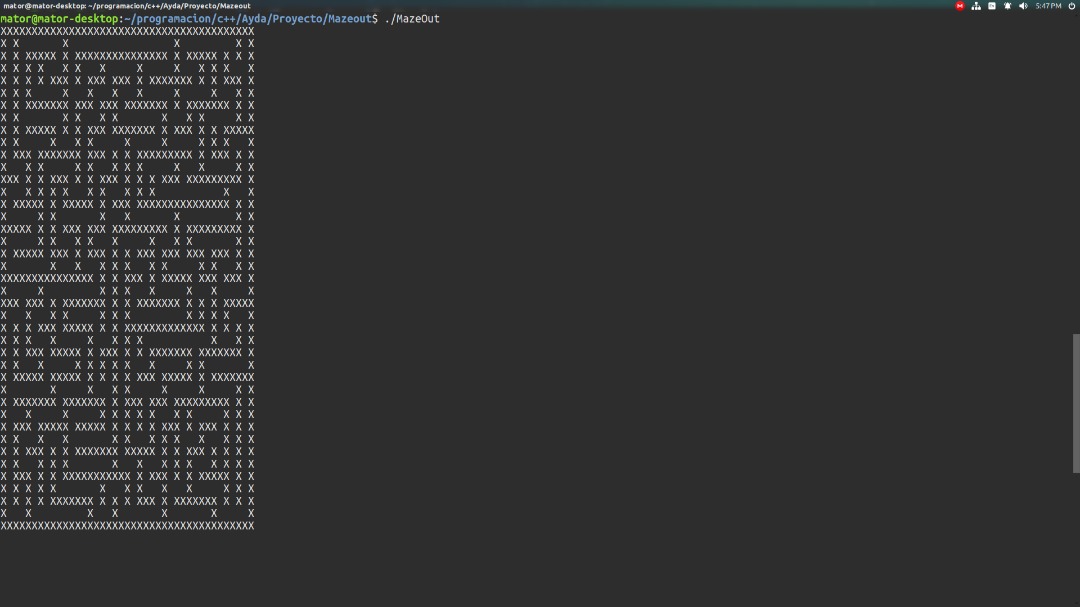
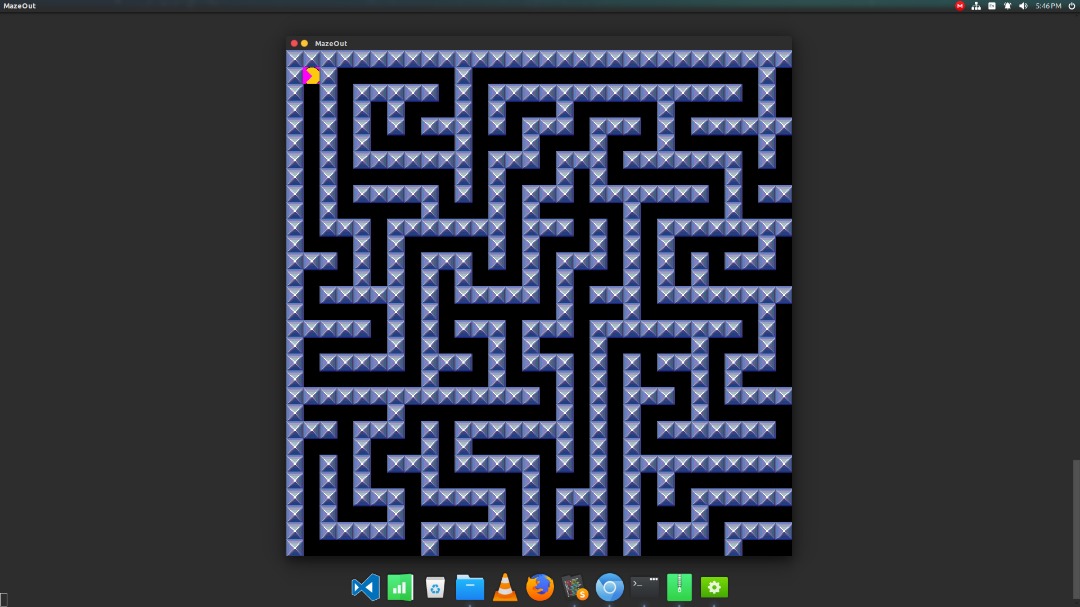
Aleph-w es una biblioteca de tipos de datos abstractos y algoritmos en torno a una amplia gama de estructuras de datos y problemas fundamentales de computación. Está programada en C y C++, y puede usarse para estos lenguajes así también como para otros empleando las adecuadas convenciones de encadenamiento.

**4.4.- La interfaz gráfica**

Una interfaz gráfica presenta un mecanismo amigable al usuario para interactuar con un programa. De esta manera, se reduce el tiempo que se requiere para aprender a usar un programa aumentando la habilidad para usarlo de una manera productiva.

La interfaz gráfica de usuario es el componente que permite la ejecución del juego, utilizando un conjunto de imágenes y objetos gráficos para representar la historia y las acciones que se van desarrollando con la dinámica del juego.

Para desarrollar la interfaz gráfica de MazeOut se usó **Allegro**, una biblioteca libre y de código abierto para la programación de videojuegos desarrollada en lenguaje C. La biblioteca cuenta con funciones para gráficos, manipulación de imágenes, texto, sonidos, dispositivos de entrada (teclado, ratón y mandos de juego) y temporizadores, así como rutinas para aritmética de punto fijo y acceso al sistema de archivos.

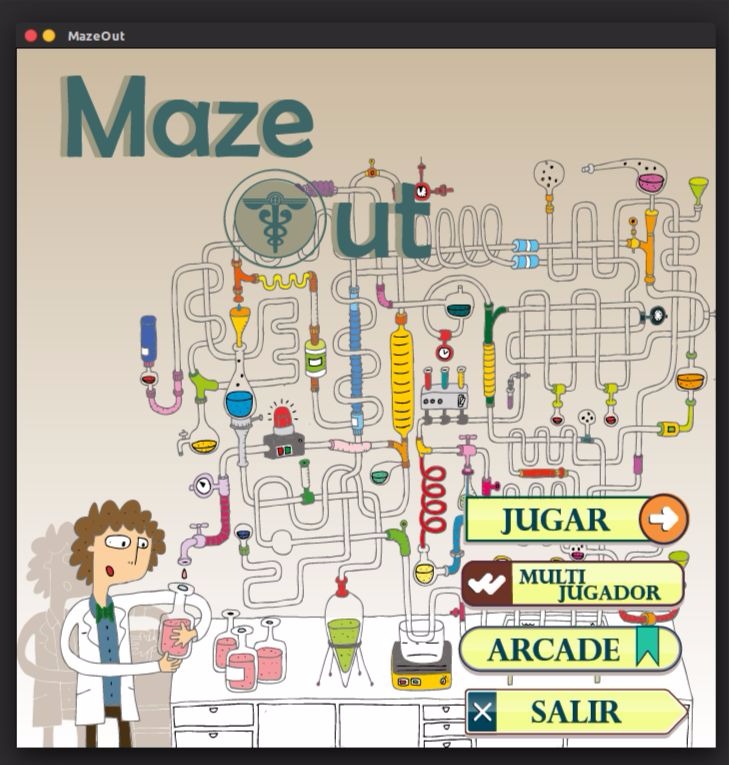


Laberinto aleatorio generado en consola Laberinto generado posteriormente dibujado con Allegro

La metodología para el desarrollo de software es la forma sistemática de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito, permite definir qué hacer, como y cuando durante todo el desarrollo y mantenimiento del proyecto. La metodología comprende los procesos a seguir sistemáticamente para idear, implementar y mantener un producto software desde que surge la necesidad del producto hasta que se cumple el objetivo por el cual fue creado.

**4.5.- Componentes del Video juego**

* **Movilidad de la unidad principal:**
  + Control de la unidad desde teclado.
* **Enemigos:** 
  + Inteligencia de la unidad.
  + Movilidad de la unidad (Deberá recorrer el mejor camino hacia el usuario).
* **Menú principal y menú de pausa:**
  + Diseños para el sistema de menús.
* **Modo de juego: Arcade**
  + Funcionamiento del modo de juego Arcade.
* **Modo de juego: Single-player**
  + Trama principal del modo de juego Single-player
* **Modo de juego: Local multi-player (Pantalla dividida)**
  + Funcionamiento del modo de juego local multi-player

****

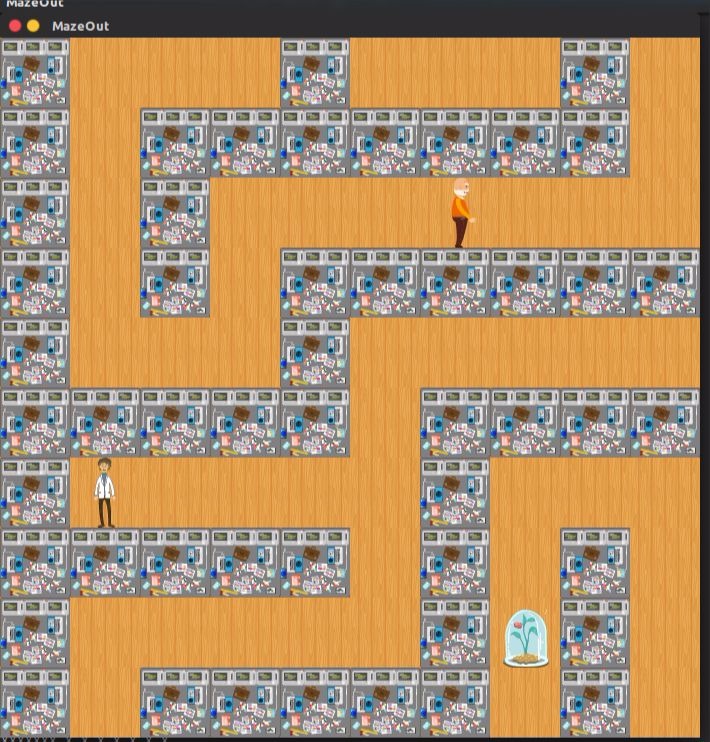
**Menú principal del juego**

****

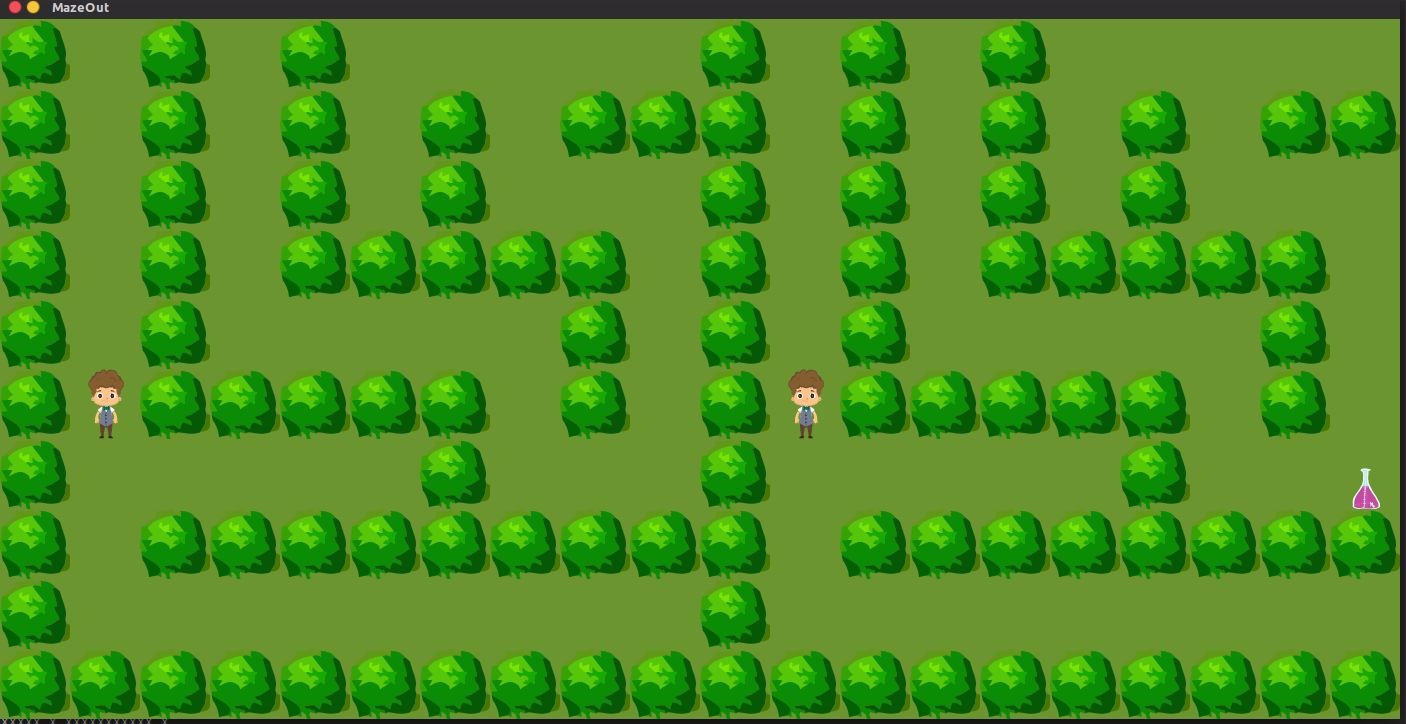
**Nivel 1: Resolver el laberinto evitando que los criminales te encuentren**

****

**Mensaje de pérdida en el nivel 1**

****

**Nivel 2: Hallar los recursos para generar la formula y usarla en los primeros humanos**

****

**Pantalla de multijugador**

**Referencias Bibliográficas**

León, Leandro. (2011). Tejiendo algoritmos: Búsqueda avanzada y grafos. Tomo II. Mérida, Venezuela. Universidad de Los Andes. Consejo de Publicaciones.

Wills, Fernando y otros (2006). Círculo, Enciclopedia Universal. Bogotá: Editorial Circulo de Lectores.