**Instituto Tecnológico de Costa Rica**

**Ingeniería en Computación**

**Profesor: Jean Carlos Miranda Fajardo**

**Proyecto #2**

***Laberinto***

Un laberinto es una matriz bidimensional con n número de filas y m número de columnas en donde cada celda tiene un valor entero para representar camino (0), pared (1), inicio del laberinto (-1) y la meta del laberinto (2). A continuación se muestra un ejemplo de los valores que tendría la estructura matriz y como se representaría a nivel gráfico.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| -1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 2 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figura 1: Ejemplo de la Estructura de la matriz**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**Figura 2: Ejemplo de visualización del laberinto**

**tomando la estructura de la figura 1**

La matriz deberá tomarse de archivos de texto plano (.txt) y generar las estructuras correspondientes en su aplicación de tal forma que pueda representarse la interfaz de la misma.

## ¿Qué se busca con este proyecto?

El objetivo general de este proyecto es facilitar un acercamiento con el juego de laberintos desde la perspectiva de la programación, de manera que las partes del juego y las reglas que lo enmarcan sean analizadas minuciosamente para desarrollar un diseño que sea programado mediante el lenguaje Python. Con esto se busca:

1. Practicar las habilidades de aplicaciones de software.
2. Ejercitar la toma de decisiones sobre el dominio del problema y de la solución.
3. Aplicar los conceptos de programación, manejo de archivos y de matrices.
4. Manejo de componentes para el uso de interfaz gráfica.

## Proyecto a desarrollar

Este proyecto, aunque no requiere de un conocimiento profundo en cuanto a estrategias, sí requiere de un conocimiento detallado del juego de laberinto. Antes de iniciar el trabajo debe comprender por completo el funcionamiento del juego, ya que lo que se desea es reproducir el comportamiento del mismo.

Asegúrese de conocer a la perfección:

1. El modo de juego
2. Movimientos
3. Dimensiones
4. Finalizaciones del juego
   1. Gane
   2. Time’s Up (Contra tiempo)
   3. Abandono

El juego debe disponer de las siguientes funcionalidades:

**Ventana de inicio:**

Se debe desarrollar una ventana en la cual el usuario pueda elegir entre dos opciones:

1. Iniciar Juego
2. Ver ranking

**Iniciar Juego:**

Debe desarrollar la funcionalidad para generar un nuevo juego, donde el usuario podrá personalizar el juego por medio de las siguientes opciones:

1. Seleccionar el tipo de juego:
   1. Normal: el usuario resuelve el juego durante el tiempo que desee
   2. Contra tiempo: se le brindan al usuario 5 minutos para que resuelva el juego, una vez terminado se le bloquea (time’s up) y pierde.
2. Seleccionar las dimensiones: El juego tendrá niveles de juego y esto serán de acuerdo a las dimensiones de la matriz:
   1. Nivel Fácil, matrices de mínimo 8 filas por 8 columnas y máximo 10 filas y 10 columnas.
   2. Nivel Medio, matrices de mínimo 11 filas por 11 columnas y máximo 13 filas por 13 columnas.
   3. Nivel Difícil, matrices de mínimo 14 filas por 14 columnas.
3. Iniciar: se le muestra al usuario el área de juego. Debe cargarse el laberinto desde el txt (seleccionado por el usuario) y desplegarlo en interfaz según lo indicado en su contenido.

**Área de juego:**

El área de juego debe disponer de lo siguiente:

1. Un área para poder jugar, donde inicialmente se muestra el laberinto y el “tótem” al inicio del laberinto y el usuario pueda ir moviéndolas un espacio válido a la vez.
2. Cronómetro: según corresponde, en modo de juego normal, va sumando segundos mostrando el tiempo invertido; en el modo contratiempo va mostrando el tiempo que le va quedando para que se acabe el tiempo.
3. Botón de abandono: termina el juego y muestra la ventana de estadísticas de juego (ver abajo).
4. Botón de reinicio: devuelve el juego al estado inicial de las piezas y reinicia el tiempo.
5. Botón de auto solución: muestra la solución del laberinto
6. Indicar gane: después de cada movimiento debe validar si el usuario ganó, si es el caso indicarlo y mostrar la ventana de estadísticas de juego (ver abajo). Se debe solicitar el nombre para el recuento del ranking.

**Reiniciar:**

Permite regresar el juego a su estado inicial.

**Auto solución:**

Debe mostrar en la ventana de juego, de manera gráfica, el paso a paso de la solución del laberinto, debe ir mostrando el movimiento del “tótem” siguiendo el camino correcto

**Estadísticas por juego:**

Por cada juego se debe indicar lo siguiente:

1. Nombre (en caso de gane)
2. Tiempo invertido
3. Dimensiones
4. Cantidad de movimientos
5. Resultado

**Ranking:**

Se debe mostrar información de los 10 juegos con mejores tiempos (de los juegos que se ganaron). Se deben ordenar de menor a mayor tiempo. Por cada juego se debe mostrar la siguiente información:

1. Tiempo invertido
2. Nombre
3. Dimensiones
4. Cantidad de movimientos

## Aspectos técnicos

El proyecto deberá estar escrito en el lenguaje de programación Python y se deben desarrollar las funcionalidades de usuario por medio de interfaz gráfica.

Se debe poder navegar siempre a la ventana inicial del juego.

Y considerar lo siguiente:

* Se deben manejar mensajes claros al usuario.
* Realizar validaciones de captura de campos y movimientos.
* Toda función bult-in de Python que deseen utilizar debe ser validada con el profesor (no incluye las relacionadas a interfaz).
* En el desarrollo del programa se deberá utilizar **iteración**.
* Deben utilizar nombres de variables, argumentos y funciones significativas.

## Documentación

El código fuente debe tener documentación interna, con comentarios precisos y bien ubicados. Cada función debe tener descripción, entradas, salidas y restricciones.

La documentación es un aspecto de gran importancia en el desarrollo de programas, especialmente en tareas relacionadas con el mantenimiento de los mismos.

Para la documentación interna, deberán incluir comentarios descriptivos para cada función, con sus entradas, salidas, restricciones y **objetivo**.

La documentación externa deberá incluir:

1. Portada.
2. Tabla de contenidos.
3. Manual de usuario.
4. Pruebas de funcionalidad: incluir *screenshots*.
5. Descripción del problema.
6. Librerías usadas: creación de archivos, etc.
7. Análisis de resultados: objetivos alcanzados, objetivos no alcanzados, y razones por las cuales no se alcanzaron los objetivos (en caso de haberlos).
8. Utilizar un repositorio de control de versiones (GIT)
9. Conclusión (es)

## Evaluación

La evaluación se va a centrar en dos elementos: programación y documentación.

El proyecto programado tiene un valor de **20%** de la nota final, en el rubro de Proyectos.

Desglose de la evaluación del proyecto programado:

1. Documentación interna 4 ptos.
2. Documentación externa 6 ptos.
3. Funcionalidad 75 ptos (ver detalle en Software a Desarrollar)
4. Revisión del proyecto (según completitud del proyecto y gestión del tiempo) 15 ptos

# Forma de trabajo

El trabajo se debe realizar de forma grupal

## Aspectos administrativos

Debe crear un archivo **.zip** (“P2\_integrantes.zip”) que contenga únicamente 2 carpetas llamadas **documentacion** y **programa**, en la primera deberá incluir los documentos de *word* (no pdf) solicitados, y en la segunda los archivos y/o carpetas necesarias para la implementación de esta tarea

## Entrega

Deberá subir el archivo antes mencionado al TEC Digital en el curso de PRINCIPIOS DE PROGRAMACIÓN I, en la asignación llamada “Proyecto II” debajo del rubro de “Proyectos”. Hora de entrega **martes 2 de setiembre a las 10:00 pm**

Después de esta fecha y hora, **NO SE ACEPTARÁN** más trabajos.

Los archivos fuentes pueden ser revisados en el sistema de Control de Plagio del TEC Digital. **Todo el código de cada proyecto debe ser 100% original y en caso de plagio se asignará nota cero.**