Facultad de Ingeniería-UNAM.

Nombre de la materia.
Fundamentos de programación.

Semestre: 2025-1.

Prof: Manuel Castañeda.

Título del Trabajo.
Proyecto, juego laberinto.

Nombre de los estudiantes.
Velasco Molina Ricardo Alonso.
Silvestre Morales Gael Octavio.

Grupo.
18.

Fecha de entrega.
15 de noviembre de 2024.

Laberinto en c++.

Nuestro proyecto se centra en la creación de un juego de laberinto donde el jugador debe encontrar la salida evitando tocar las paredes. Esta iniciativa resulta esencial para comprender elementos fundamentales de programación como el uso de arrays, estructuras, funciones y gestión de datos.

La aplicación ofrece funcionalidades de registro de usuarios, autenticación, niveles progresivos y seguimiento estadístico. Cada escenario se plasma en una matriz donde el jugador se desplaza mediante las teclas W, A, S, D. Entre los propósitos destacan la implementación correcta de estructuras de datos matriciales, el procesamiento de texto para credenciales, control temporal durante la partida y administración de almacenamiento tanto en memoria como en archivos.

La implementación se organizó en etapas estratégicas. Los escenarios se construyen mediante matrices bidimensionales donde: 1 representa muros, 0 espacios transitables y 2 la meta. Esta representación optimiza la validación de movimientos y manipulación del mapa. Los datos de jugadores se almacenan en estructuras que contienen nombre, clave y métricas de desempeño, permitiendo una gestión eficiente sin duplicidad.

La arquitectura se segmentó en funciones específicas para mejor mantenimiento. Por ejemplo, la autenticación verifica credenciales contra registros existentes, mientras el registro incorpora nuevos jugadores. La mecánica principal controla la visualización, validación de movimientos y actualización de estadísticas.

Para optimizar recursos, se implementó asignación dinámica de memoria. Los laberintos y el registro de usuarios se crean con dimensiones flexibles mediante punteros y malloc, permitiendo escalabilidad. Conforme crece la base de usuarios, realloc ajusta el espacio necesario.

La persistencia de datos se logra mediante archivos de texto que preservan información y estadísticas entre sesiones. Las operaciones de archivo (fopen, fread, fwrite, fclose) gestionan el guardado y recuperación de datos, manteniendo el progreso de cada jugador.

La interfaz consiste en un sistema basado en texto que presenta opciones de registro, acceso, juego y visualización de estadísticas. La función de limpieza de pantalla mejora la experiencia visual, mientras mensajes informativos guían al usuario durante la interacción.

Análisis.

El proyecto generó aprendizajes valiosos sobre gestión de datos interactivos y algoritmos en programación estructurada. Se logró implementar exitosamente todas las funcionalidades previstas: gestión de usuarios, validación, navegación por niveles y seguimiento estadístico. El cronómetro y detección de colisiones funcionan adecuadamente, garantizando una experiencia fluida.

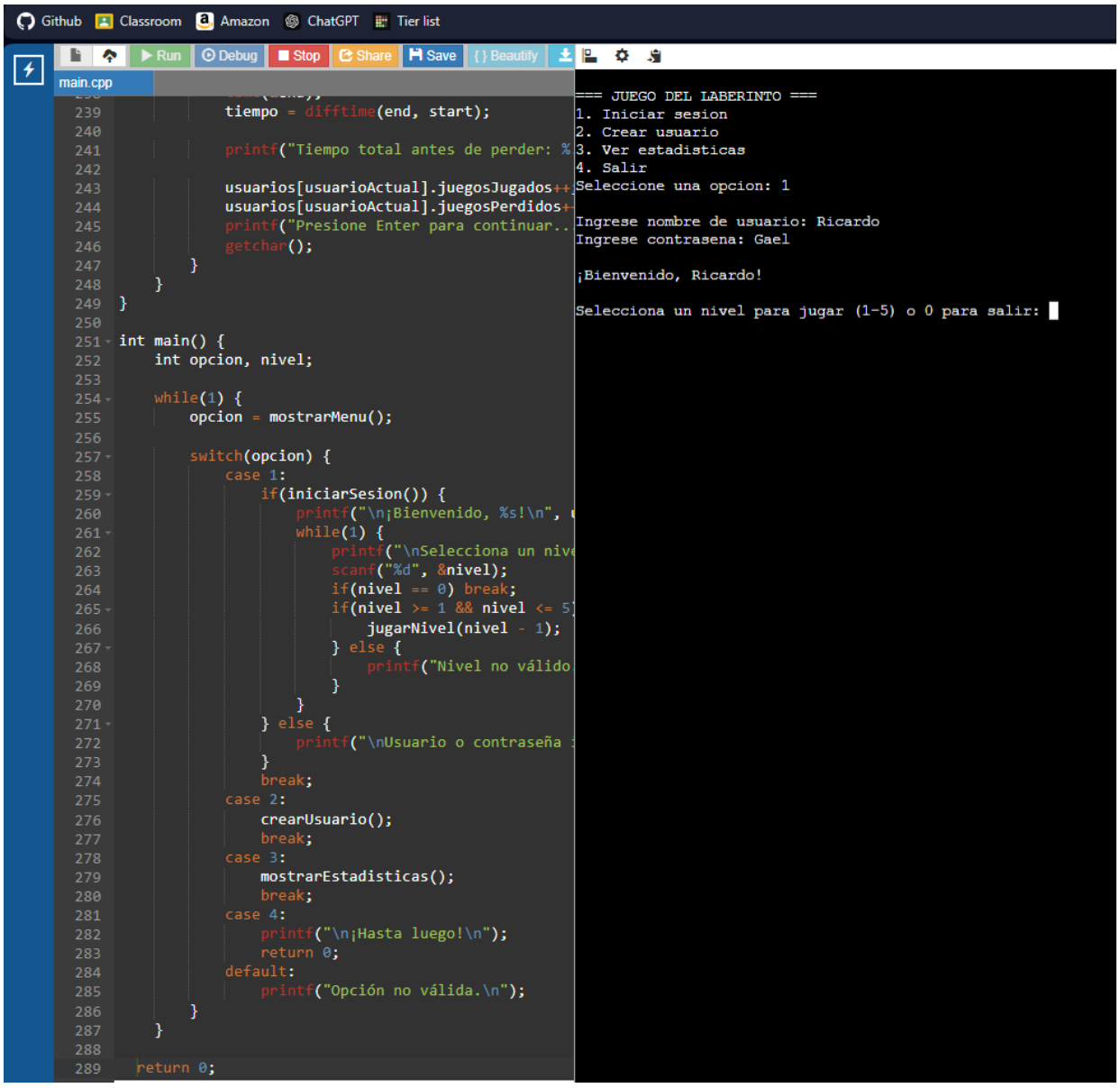
La combinación de arrays y estructuras facilitó el manejo de datos, mientras la modularización mejoró el desarrollo. La interfaz textual, aunque minimalista, cumple efectivamente su propósito comunicativo.

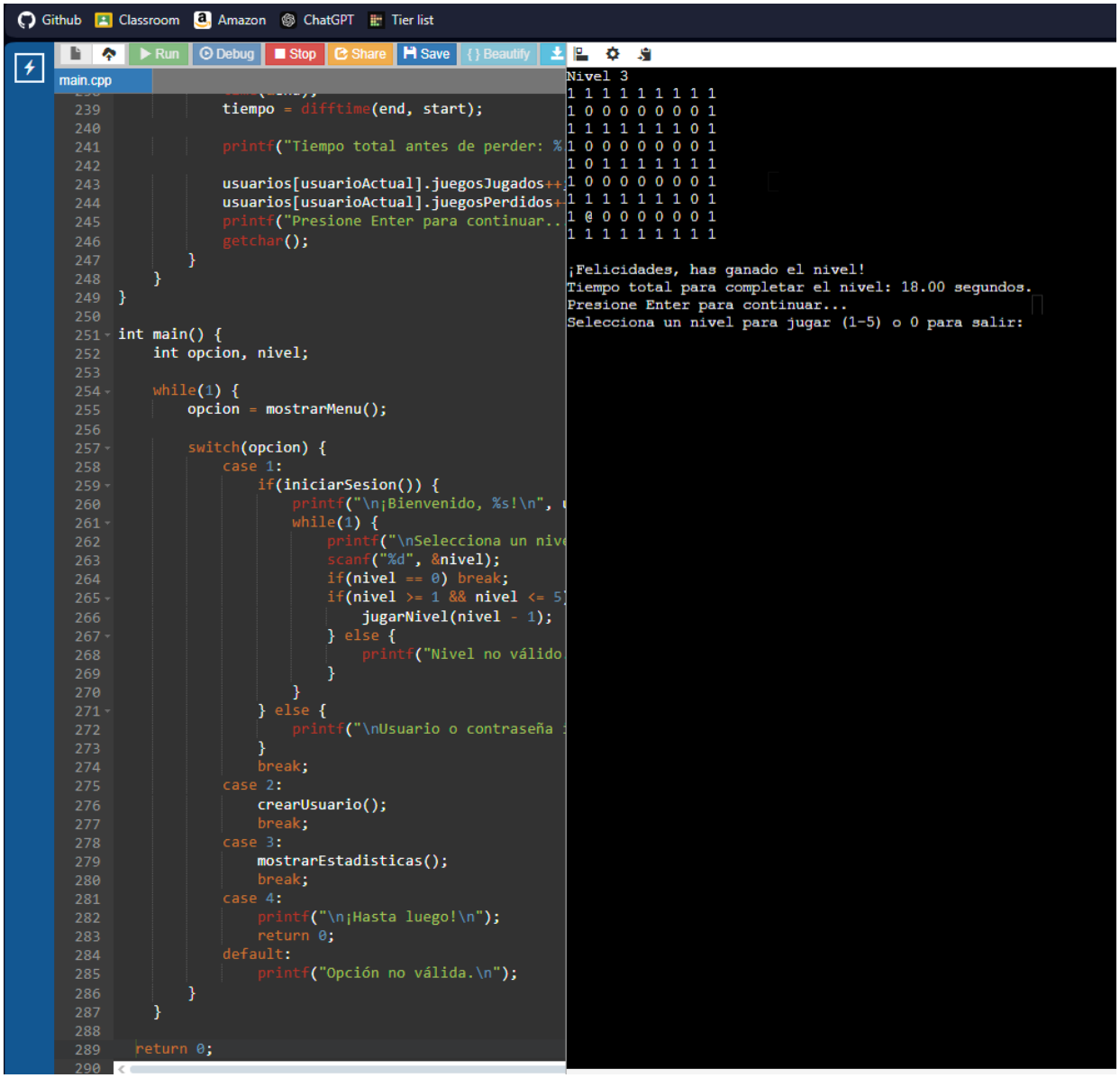
Conclusiones:

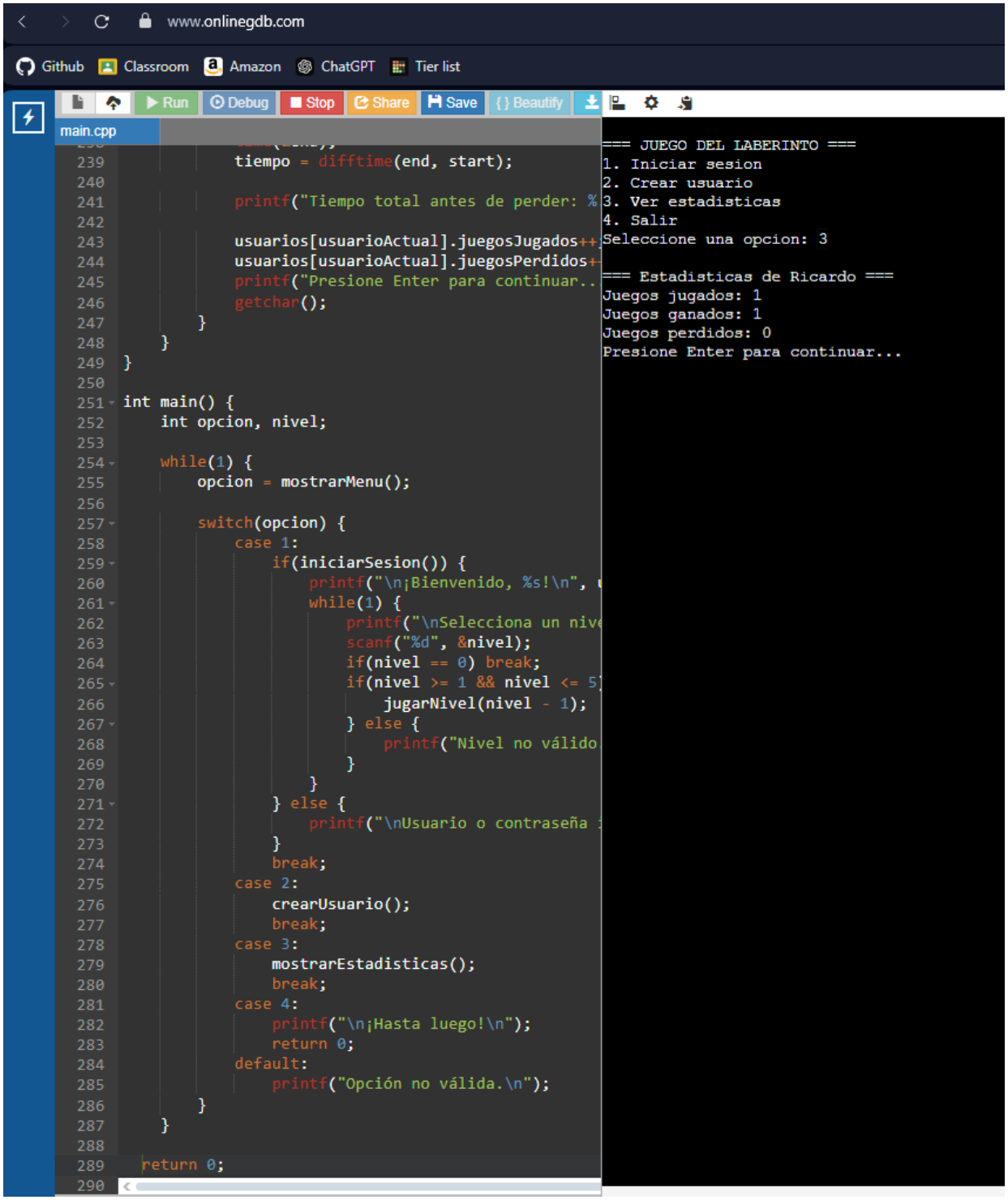
Gael: Se hizo una implementación exitosa de estructuras de datos y funciones modulares para gestión de usuarios y mecánicas de juego. La coordinación entre validación de movimientos y actualización de estados requirió múltiples iteraciones. Establecimos sistema robusto de verificación de coordenadas y actualización de datos mediante pruebas sistemáticas.

Ricardo: Hubo un desarrollo efectivo del sistema de almacenamiento y recuperación de datos de usuario. Tuvimos demasiadas complicaciones, a pesar de esto se tuvo que mantener consistencia en las estadísticas durante sesiones prolongadas. El apoyo de los temas vistos y comentarios de nuestros compañeros fueron de mucha ayuda para concluir con el trabajo.

Codigo.


Github
Classroom
Amazon
ChatGPT
Tier list
Run
Debug
Stop
Share
Save
{ } Beautify
main.cpp
tiempo = difftime(end, start);
printf("Tiempo total antes de perder: %
usuarios[usuarioActual].juegosJugados++;
usuarios[usuarioActual].juegosPerdidos++;
printf("Presione Enter para continuar..
getchar();
}
}
}
int main() {
int opcion, nivel;
while(1) {
opcion = mostrarMenu();
switch(opcion) {
case 1:
if(iniciarSesion()) {
printf("\n¡Bienvenido, %s!\n",
while(1) {
printf("\nSelecciona un nive
scanf("%d", &nivel);
if(nivel == 0) break;
if(nivel >= 1 && nivel <= 5
jugarNivel(nivel - 1);
} else {
printf("Nivel no válido
}
}
} else {
printf("\nUsuario o contraseña
}
break;
case 2:
crearUsuario();
break;
case 3:
mostrarEstadisticas();
break;
case 4:
printf("\n¡Hasta luego!\n");
return 0;
default:
printf("Opción no válida.\n");
}
}
return 0;
=== JUEGO DEL LABERINTO ===
1. Iniciar sesion
2. Crear usuario
3. Ver estadisticas
4. Salir
Seleccione una opcion: 1
Ingrese nombre de usuario: Ricardo
Ingrese contrasena: Gael
¡Bienvenido, Ricardo!
Selecciona un nivel para jugar (1-5) o 0 para salir:


Github
Classroom
Amazon
ChatGPT
Tier list
Run
Debug
Stop
Share
Save
Beautify
main.cpp
tiempo = difftime(end, start);
printf("Tiempo total antes de perder: %
usuarios[usuarioActual].juegosJugados++;
usuarios[usuarioActual].juegosPerdidos+
printf("Presione Enter para continuar..
getchar();
}
}
}
int main() {
int opcion, nivel;
while(1) {
opcion = mostrarMenu();
switch(opcion) {
case 1:
if(iniciarSesion()) {
printf("\n¡Bienvenido, %s!\n",
while(1) {
printf("\nSelecciona un nive
scanf("%d", &nivel);
if(nivel == 0) break;
if(nivel >= 1 && nivel <= 5
jugarNivel(nivel - 1);
} else {
printf("Nivel no válido
}
}
} else {
printf("\nUsuario o contraseña
}
break;
case 2:
crearUsuario();
break;
case 3:
mostrarEstadisticas();
break;
case 4:
printf("\n¡Hasta luego!\n");
return 0;
default:
printf("Opción no válida.\n");
}
}
return 0;
Nivel 3
1 1 1 1 1 1 1 1 1
1 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 0 1
1 0 0 0 0 0 0 0 1
1 0 1 1 1 1 1 1 1
1 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 0 1
1 @ 0 0 0 0 0 0 1
1 1 1 1 1 1 1 1 1
¡Felicidades, has ganado el nivel!
Tiempo total para completar el nivel: 18.00 segundos.
Presione Enter para continuar...
Selecciona un nivel para jugar (1-5) o 0 para salir:


www.onlinegdb.com
Github
Classroom
Amazon
ChatGPT
Tier list
Run
Debug
Stop
Share
Save
{ } Beautify
main.cpp
tiempo = difftime(end, start);
printf("Tiempo total antes de perder: %
usuarios[usuarioActual].juegosJugados++;
usuarios[usuarioActual].juegosPerdidos+
printf("Presione Enter para continuar..
getchar();
}
}
}
int main() {
int opcion, nivel;
while(1) {
opcion = mostrarMenu();
switch(opcion) {
case 1:
if(iniciarSesion()) {
printf("\n¡Bienvenido, %s!\n",
while(1) {
printf("\nSelecciona un nive
scanf("%d", &nivel);
if(nivel == 0) break;
if(nivel >= 1 && nivel <= 5
jugarNivel(nivel - 1);
} else {
printf("Nivel no válido
}
}
} else {
printf("\nUsuario o contraseña
}
break;
case 2:
crearUsuario();
break;
case 3:
mostrarEstadisticas();
break;
case 4:
printf("\n¡Hasta luego!\n");
return 0;
default:
printf("Opción no válida.\n");
}
}
return 0;
=== JUEGO DEL LABERINTO ===
1. Iniciar sesion
2. Crear usuario
3. Ver estadisticas
4. Salir
Seleccione una opcion: 3
=== Estadisticas de Ricardo ===
Juegos jugados: 1
Juegos ganados: 1
Juegos perdidos: 0
Presione Enter para continuar...

Manual de usuario.

1. Introducción

El Juego del Laberinto es un programa interactivo en C donde los jugadores deben moverse a través de laberintos de dificultad progresiva. Permite:

- Crear y gestionar usuarios.
- Registrar estadísticas de juegos ganados, perdidos y jugados.
- Jugar en cinco niveles con laberintos de tamaños incrementales.

Este manual te guiará en cada aspecto del juego.

2. Requisitos del Sistema

- Compilador C compatible (GCC recomendado).
- Entorno para ejecutar el programa (terminal en Linux, Windows CMD o GDB online).

3. Cómo Usar el Programa

3.1 Inicio

1. Ejecuta el programa.

El menú principal mostrará las siguientes opciones:

1. Iniciar sesión
2. Crear usuario
3. Ver estadísticas
4. Salir

2. Selecciona una opción ingresando el número correspondiente.

3.2 Crear Usuario

1. Selecciona la opción 2. Crear usuario.
2. Ingresa un nombre de usuario (máximo 9 caracteres).
3. Ingresa una contraseña (máximo 9 caracteres).
4. El sistema confirmará la creación del usuario.

Nota: Puedes registrar hasta 10 usuarios.

3.3 Iniciar Sesión

1. Selecciona la opción 1. Iniciar sesión.
2. Ingresa tu nombre de usuario y contraseña.
3. Si la información es correcta, accederás al sistema.

Errores comunes:

- Ingresar un nombre o contraseña incorrectos.
- Intentar iniciar sesión sin haber creado un usuario.

3.4 Ver Estadísticas

1. Selecciona la opción 3. Ver estadísticas.
2. Si has iniciado sesión, se mostrarán tus estadísticas personales:
   - Juegos jugados.
   - Juegos ganados.
   - Juegos perdidos.
3. Si no has iniciado sesión, se mostrará un mensaje indicando que debes iniciar sesión primero.

3.5 Jugar un Nivel

1. Inicia sesión y selecciona un nivel (1 a 5).
2. Aparecerá el laberinto en pantalla con los siguientes elementos:

- 0: Espacio vacío (puedes moverte aquí).
- 1: Pared (no puedes atravesarla).
- 2: Meta (tu objetivo).
- @: Tu posición actual.

3. Usa las siguientes teclas para moverte:

- W: Arriba.
- A: Izquierda.
- S: Abajo.
- D: Derecha.

4. Completa el nivel moviéndote hacia la meta (2).

- Si llegas a la meta, ganas el nivel.
- Si chocas contra una pared, pierdes y el nivel se reinicia.

5. Al finalizar el nivel (ganado o perdido), se mostrarán:

- Tiempo total jugado.
- Estadísticas actualizadas.

3.6 Finalizar el Programa

Selecciona la opción 4. Salir en el menú principal para cerrar el programa.

4. Características Técnicas

4.1 Usuarios

- Registro: Hasta 10 usuarios únicos.
- Estadísticas por usuario:
- Juegos jugados.
- Juegos ganados.
- Juegos perdidos.

4.2 Laberintos

- Niveles: Cinco niveles con tamaños y complejidad crecientes:
- Nivel 1: 5x5.
- Nivel 2: 7x7.
- Nivel 3: 9x9.
- Nivel 4: 11x11.
- Nivel 5: 13x13.
- Mapa: Matrices que contienen el diseño del laberinto.

4.3 Estadísticas

Las estadísticas de cada usuario se actualizan automáticamente tras cada partida:

- Juegos ganados: Si llegas a la meta.
- Juegos perdidos: Si chocas contra una pared.

5. Instrucciones Avanzadas

5.1 Reinicio del Nivel

Si pierdes en un nivel:

- El programa reiniciará automáticamente tu posición al inicio del nivel.
- Podrás volver a intentar completarlo.

5.2 Optimización del Juego

El diseño modular facilita modificaciones:

- Añadir niveles: Puedes agregar más laberintos a la matriz laberintos y aumentar el valor de NIVELES.
- Cambiar controles: Edita las instrucciones de movimiento en la función jugarNivel.

6. Solución de Problemas

6.1 Errores Comunes

1. Usuario no encontrado al iniciar sesión:
   - Verifica que ingresaste correctamente tu nombre y contraseña.
   - Asegúrate de haber creado el usuario.
2. Mensaje de “Nivel no válido”:
   - Asegúrate de ingresar un número entre 1 y 5.
3. No se reinicia el nivel tras perder:
   - Asegúrate de seguir las instrucciones en pantalla y presionar Enter.

6.2 Posibles Mejoras

- Implementar persistencia para guardar usuarios y estadísticas entre sesiones.
- Añadir soporte para multijugador o niveles personalizados.

7. Créditos

Este juego ha sido diseñado como un proyecto educativo en C. Permite practicar estructuras, matrices y manejo de usuarios.

Complicaciones y observaciones finales.

Debido a la falta de practica con otros compiladores nos fue muy complicado realizar la elaboración total del codigo, llegamos a presentar problemas de distintos tipos, el mayor problema que se nos presento fue la elaboración de los laberintos, así como el uso de otra librería, la cual era útil para contar el tiempo de las partidas jugadas, al igual que intentamos hacer que el movimiento del jugador se diera al momento de teclear sin la necesidad de presionar “enter”, logramos hacerlo con el menú principal, pero al no añadir más librerías se nos hizo muy complicado avanzar con el proyecto en sí. Otro problema que se nos presento fue que no logramos hacer que la sesión mantuviera sus datos de estadísticas guardadas al momento de entrar y salir del programa, por otro lado, el desarrollo de la práctica o proyecto nos fue de mucho ayuda para resolver cuestiones por el estilo de manera eficiente.