


31 DE OCTUBRE DE 2022

## PROYECTO PROGRAMADO 3

LABERINTO DIRIGIDO

JEAN HUNT  
LENGUAJES DE PROGRAMACIÓN  
Ingeniería en computación



## Tabla de contenido

Manual de usuario.....	1
<b>Compilación y ejecución.....</b>	<b>2</b>
<b>Tutorial .....</b>	<b>3</b>
1. <i>Inicio</i> .....	3
2. <i>Juego nuevo</i> .....	3
3. <i>Partida en progreso</i> .....	5
3.1. <i>Solicitar sugerencia</i> .....	6
4. <i>Partida finalizada</i> .....	7
5. <i>Estadísticas</i> .....	8
6. <i>Repeticiones</i> .....	9
Documentación general .....	10
<b>1. Descripción del problema</b> .....	<b>11</b>
<b>2. Diseño del programa:</b> .....	<b>11</b>
2.1. <i>Decisiones de diseño</i> .....	11
2.2. <i>Algoritmos usados</i> .....	12
<b>3. Librerías utilizadas</b> .....	<b>13</b>
<b>4. Análisis de resultados</b> .....	<b>13</b>

Ilustración 1: Compilación y ejecución .....	2
Ilustración 2: Inicio .....	3
Ilustración 3:Iniciar juego.....	4
Ilustración 4:Seleccionar laberinto .....	4
Ilustración 5: Tablero de juego .....	5
Ilustración 6:Sugerencia no disponible.....	6
Ilustración 7:Verificación correcta .....	6
Ilustración 8:Verificación incorrecta .....	6
Ilustración 9:Partida finalizada .....	7
Ilustración 10:Estadísticas .....	8
Ilustración 11:Repetición.....	9
Ilustración 12:Algoritmos utilizados .....	12
Ilustración 13: Objetivos alcanzados .....	14

## *Manual de usuario*

## Compilación y ejecución

Este programa es ejecutado en el lenguaje de programación Python y prolog por lo que es necesario tener instalados ambos ambientes antes de ejecutar el programa. Los respectivos enlaces para la descarga de estos lenguajes son:

- <https://www.swi-prolog.org/download/stable>
- <https://www.python.org/downloads/>

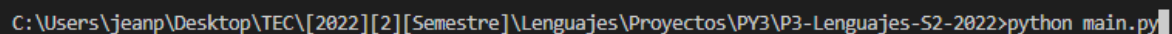
Además, es necesaria la instalación de dos librerías externas cuyos comandos de instalación se encuentran a continuación:

- `python3 -m pip install pillow`
- `pip install pyswip`

Una vez completados los pasos anteriores el programa está listo para ser ejecutado, para ello:

1. Abrir una terminal de comandos, ya sea integrada o de terceros.
2. Dirigirse a la ruta en donde se encuentra el archivo proyecto.
3. Ejecutar el programa haciendo uso del comando “Python main.py”

**Nota:** Puede compilar el programa desde cualquier directorio, pero tendrá que ingresar la ruta completa del archivo, por ejemplo: Python C://escritorio/programa/main.py



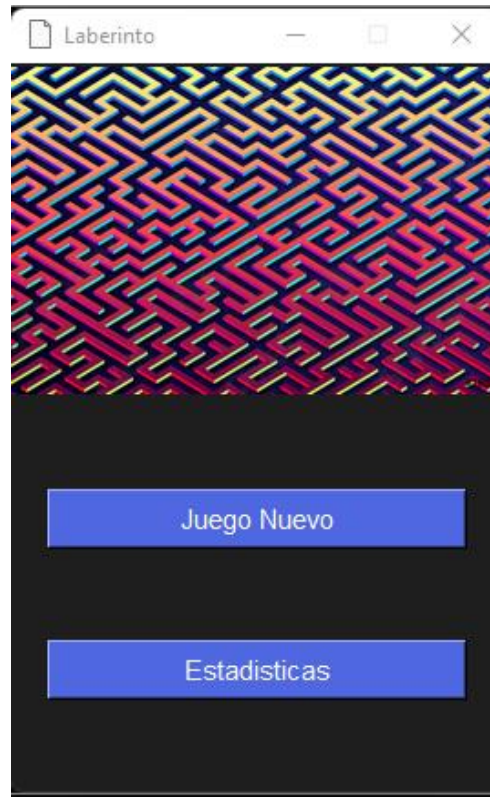
```
C:\Users\jeanp\Desktop\TEC\[2022][2][Semestre]\Lenguajes\Proyectos\PY3\P3-Lenguajes-S2-2022>python main.py
```

*Ilustración 1: Compilación y ejecución*

## Tutorial

### 1. Inicio

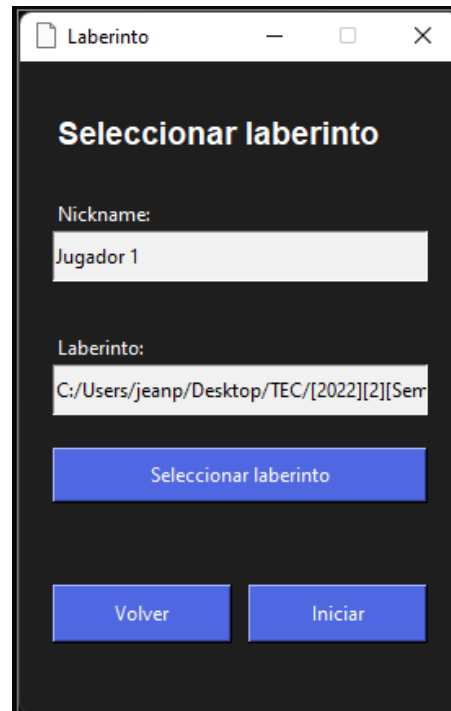
La primera pantalla que se muestra al ejecutar el programa es la pantalla de inicio o “home”, esta pantalla permite al usuario iniciar un juego nuevo o ver las estadísticas de partidas anteriores haciendo clic en el botono correspondiente.



*Ilustración 2: Inicio*

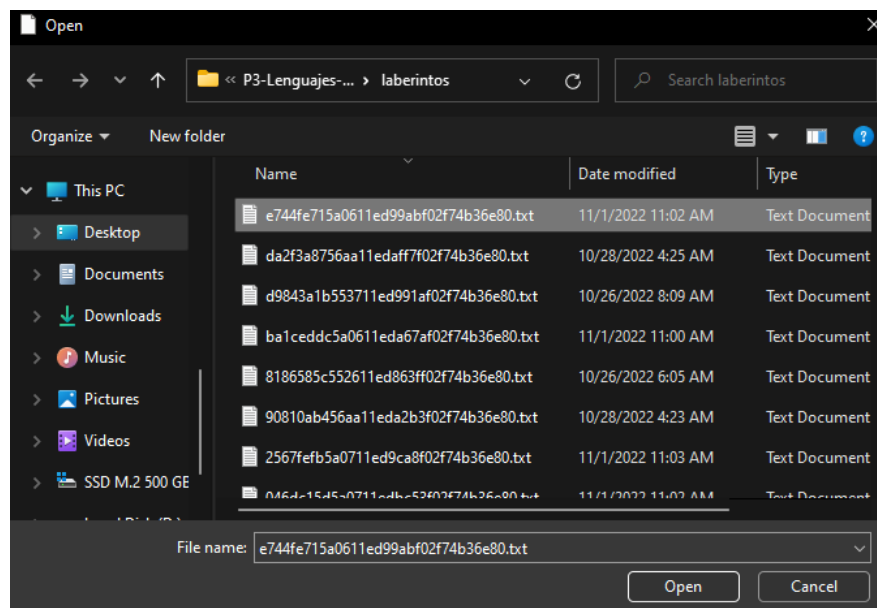
### 2. Juego nuevo

Al hacer clic en “juego nuevo” en la pantalla de inicio se le redirigirá a dicha pantalla, en la que se le solicitará un nombre de usuario y que seleccione un laberinto tal y como se muestra a continuación:



*Ilustración 3: Iniciar juego*

Al hacer clic en seleccionar laberinto se desplegará un dialogo en el que se deberá seleccionar el archivo de texto que contiene el laberinto que se quiere jugar.



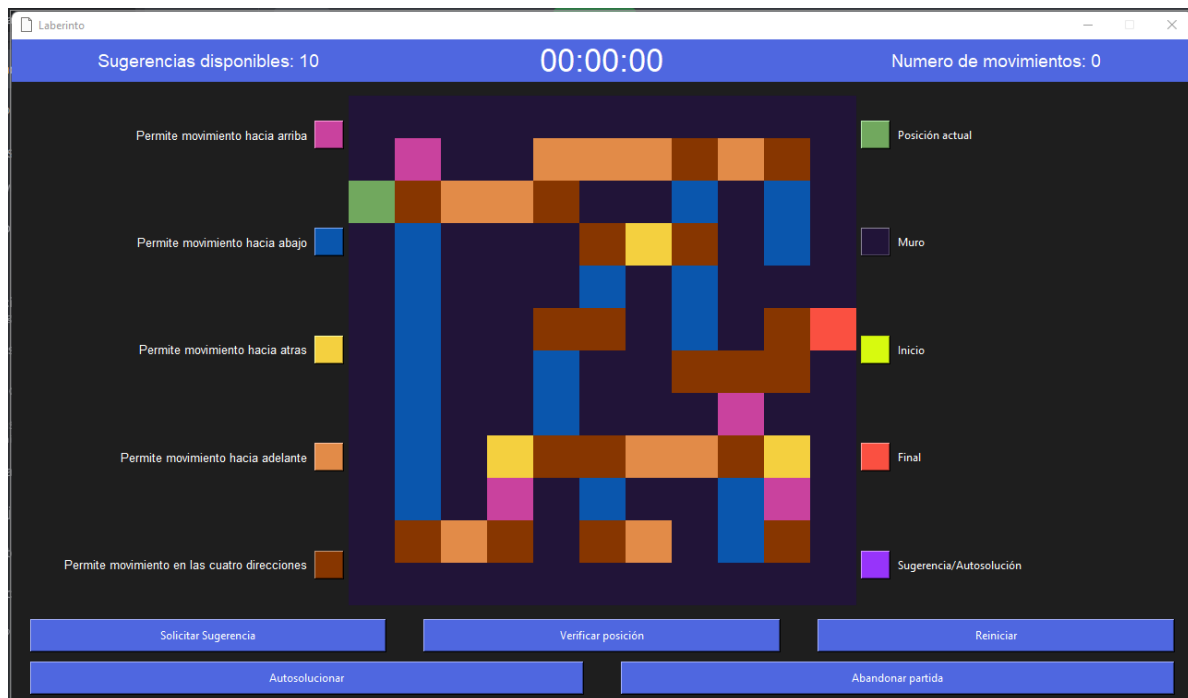
*Ilustración 4: Seleccionar laberinto*

Luego de seleccionar el laberinto que desea jugar haga clic en “iniciar” para iniciar la partida.

### 3. Partida en progreso

Al iniciar la partida, se desplegará la siguiente pantalla la cual contiene información como:

- Sugerencias disponibles
- Cronometro
- Movimientos realizados
- Tablero de juego
- Contexto del tablero
- Panel de funciones



*Ilustración 5: Tablero de juego*

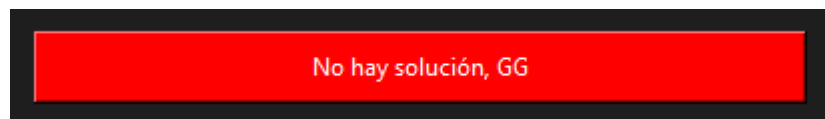
El movimiento de la ficha se llevará a cabo mediante las teclas de dirección, una vez realice el primer movimiento se iniciará el cronometro. Cada movimiento será



registrado en un contador además de que cuenta con 10 sugerencias para movimiento de la ficha.

### 3.1. Solicitar sugerencia

Al seleccionar esta opción se iluminará una casilla del tablero durante 3 segundos en caso de que exista un posible movimiento, caso contrario, se le indicará que no existe ningún movimiento posible de la siguiente manera:



*Ilustración 6: Sugerencia no disponible*

### 3.2. Verificar

Al seleccionar esta opción el programa validará si la posición en la que se encuentra actualmente forma parte de la solución del laberinto, de ser así, esto se verá reflejado de la siguiente manera:



*Ilustración 7: Verificación correcta*

Caso contrario:



*Ilustración 8: Verificación incorrecta*

### 3.3. Reiniciar

Devuelve el laberinto a su estado inicial.

### 3.4. Auto solución

Se le mostrará la solución del laberinto actual, sin embargo, esto contará como abandono de partida.

### 3.5. Abandonar partida

Se abandona la partida.

## 4. Partida finalizada

Al finalizar la partida se le mostrará una pantalla con información como:

- Sugerencias utilizadas
- Movimientos realizados
- Nickname
- Tipo de finalización
- Laberinto jugado
- Tiempo
- Panel de opciones



Ilustración 9: Partida finalizada

#### 4.1. Volver a inicio

Vuelve a la página inicial del programa.

#### 4.2. Guardar repetición

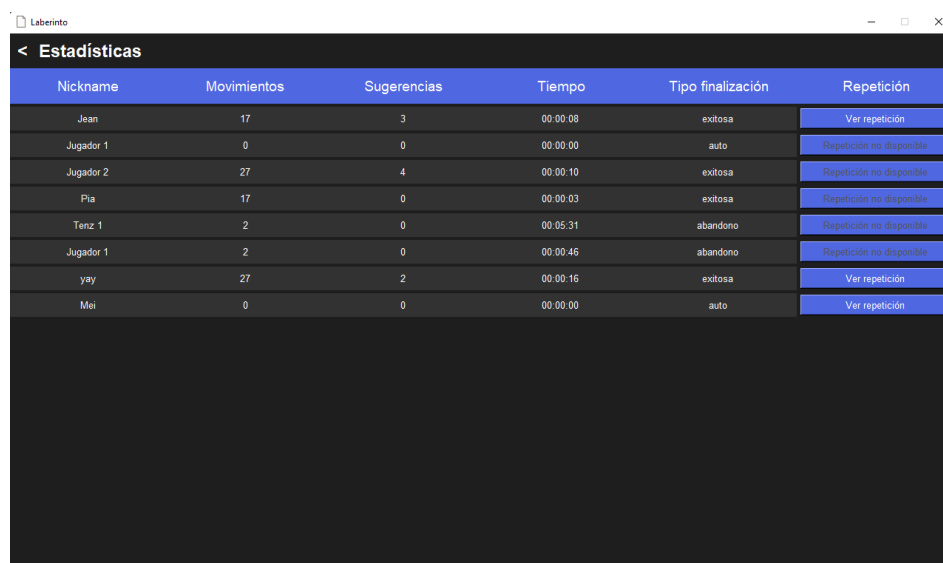
Guarda una repetición de la partida recién finalizada. Esta podrá ser reproducida en el apartado de estadísticas del programa.

### 5. Estadísticas

Cada partida finalizada o abandonada es almacenada para mantener un registro estadístico de las mismas. Cuando un usuario termina una partida se le da la opción de “guardar repetición”. Estas repeticiones pueden ser accedidas desde esta ventana.

Las estadísticas contienen información como:

- Nickname
- Movimientos realizados
- Sugerencias solicitadas
- Tiempo
- Tipo finalización
- Repetición (si existe)



Nickname	Movimientos	Sugerencias	Tiempo	Tipo finalización	Repetición
Jean	17	3	00:00:08	exitosa	Ver repetición
Jugador 1	0	0	00:00:00	auto	Repetición no disponible
Jugador 2	27	4	00:00:10	exitosa	Repetición no disponible
Pia	17	0	00:00:03	exitosa	Repetición no disponible
Tenz 1	2	0	00:05:31	abandono	Repetición no disponible
Jugador 1	2	0	00:00:46	abandono	Repetición no disponible
yay	27	2	00:00:16	exitosa	Ver repetición
Mei	0	0	00:00:00	auto	Ver repetición

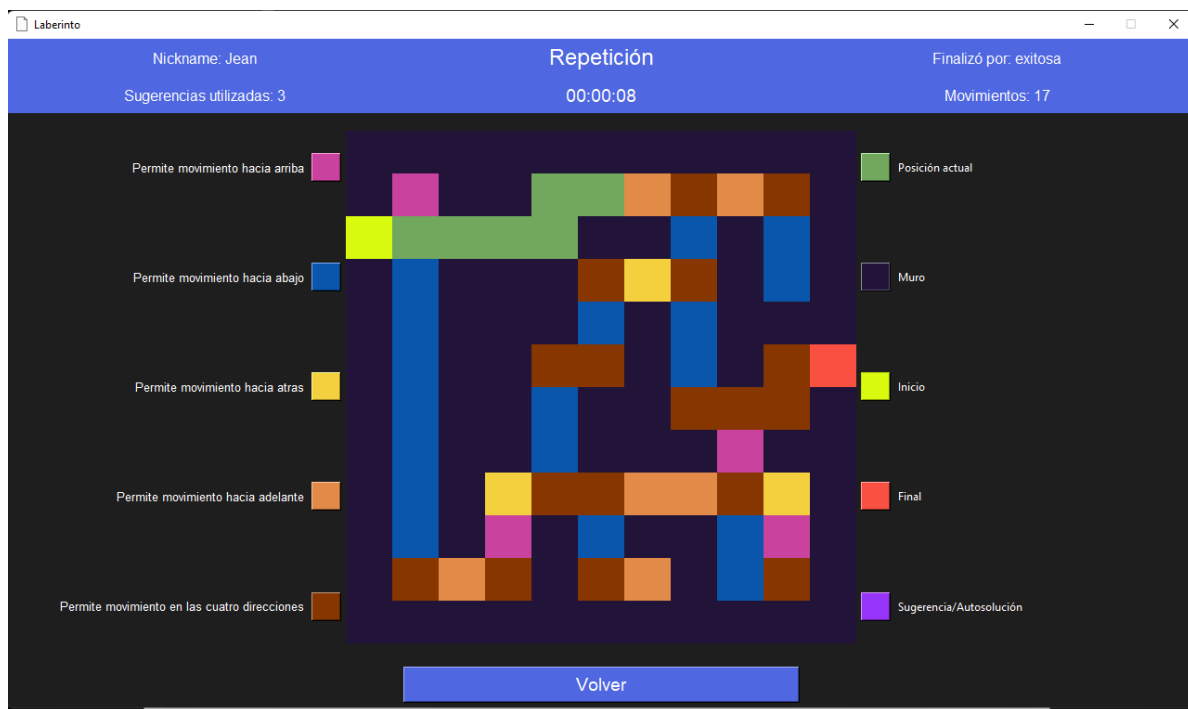
Ilustración 10: Estadísticas

## 6. Repeticiones

Reproduce una repetición de una partida, es decir, muestra paso a paso los movimientos de un jugador en una partida, sugerencias solicitadas y si este solicitó auto solución.

Esta ventana contiene información como:

- Nickname
- Sugerencias utilizadas
- Movimientos
- Tiempo
- Tipo finalización
- Tablero del laberinto jugado
- Contexto del tablero



*Ilustración 11: Repetición*

## *Documentación general*

## 1. Descripción del problema

Se deberá diseñar y desarrollar una aplicación que permite jugar un laberinto dirigido, es decir, laberintos que solo permite movimientos en direcciones específicas dada la posición. Además de funcionalidades como sugerencias, auto soluciones, verificación de posición, repeticiones y estadísticas.

La parte aplicativa deberá ser desarrollada en algún programa que facilite este proceso, sin embargo, la parte lógica deberá ser desarrollada en prolog.

## 2. Diseño del programa:

El programa fue desarrollado de manera hibrida entre Python y prolog, siendo prolog la parte lógica y Python la parte aplicativa.

La estructura del proyecto es la siguiente:

- **Main.py:** Contiene la parte aplicativa del programa
- **Scroll.py:** Permite crear frames con scroll
- **Logica.pro:** Contiene la parte lógica del programa
- **Repeticiones:** Contiene los archivos con la información de las repeticiones
- **Laberintos:** Contiene los laberintos que se han jugado
- **Img:** Contiene las imágenes utilizadas en el programa

### 2.1. Decisiones de diseño

Antes y durante el desarrollo del proyecto se tomaron las siguientes decisiones:

- La parte aplicativa del programa será desarrollada en lenguaje Python
- Se utilizará pyswip como intermediario entre Python y prolog
- Se utilizará tkinter para el diseño de la interfaz grafica
- Las repeticiones serán almacenadas en archivos txt de manera individual

- Se creará una copia del laberinto jugado para evitar fallos al momento de repeticiones
- Se utilizará vscode como entorno de desarrollo.
- Se utilizará GitHub como control de repositorio.
- Las repeticiones y los laberintos tendrán el mismo nombre para facilitar su acceso.

## 2.2. Algoritmos usados

A continuación, se encuentra un listado con las declaraciones de los algoritmos utilizados en el programa:

```

1  def getVentana()
2  def cambiarTama(x, y)
3  def raise_frame(frame,x,y)
4  def crearPaginaInicio()
5  def crearPaginaPreJuego()
6  def crearPaginaTablero()
7  def crearPaginaFinal()
8  def crearPaginaEstadisticas(ventanaAnterior)
9  def getEstadisticas()
10 def getRepeticion(id)
11 def getPaginaRepeticion(id)
12 def reproducirRepeticion(tablero, repeticion)
13 def reproducirRepeticionAux(tab,punto)
14 def parpadear(objeto, col1, col2, cant)
15 def reiniciar()
16 def guardarEstadisticas(pNickname, pCantmov, pCantSug,pTiempo, pTipoFin)
17 def guardarRepeticion(id)
18 def crearFrame()
19 def solicitarArchivo()
20 def reestablecerValores ()
21 def iniciarJuego()
22 def transformarLaberinto(laberintoProlog)
23 def obtenerPosicionInicial()
24 def abandonarPartida()
25 def solucionarLaberinto(laberintoSol, puntoInicio, puntoFinal)
26 def solicitarSugerencia(boton)
27 def mostrarSugerencia(ficha)
28 def crearTablero(ventanaTablero,laberintoTablero)
29 def autoSolucionar(boton)
30 def mostrarSolucion(tab)
31 def obtenerSolucion()
32 def verificar(boton)
33 def verificarAux(boton)
34 def moverFichaAux(fichaActual, fichaSiguiete)
35 def moverFicha[i]
36 def moverIzquierda(event)
37 def moverDerecha(event)
38 def moverArriba(event)
39 def moverAbajo(event)
40 def formatearTiempo(segundos)
41 def getTiempo()
42 def refrescarTiempo()

```

Ilustración 12:Algoritmos utilizados

### 3. Librerías utilizadas

Para el desarrollo del programa se utilizaron las siguientes librerías:

- **Os:** Validar que un archivo existe
- **Tkinter:** Desarrollo de interfaz gráfica
- **Pyswip:** Conexión entre Python y prolog
- **Pil:** Manipulación de imágenes
- **Copy:** Copy de valores a variables sin afectar su referencia
- **Datetime:** Calculo del tiempo transcurrido para el cronometro
- **Uuid:** Creación de identificadores universales únicos

### 4. Análisis de resultados

En la siguiente tabla se muestra una lista de objetivos para los cuales se evaluará si fueron cumplidos o no, en caso de no cumplirse también se indicará la razón del porqué.

Objetivo	Alcanzado	Razón
Establecer una conexión entre Python y prolog	Si	
Conseguir un diseño agradable y moderno de UI	No	Tkinter es muy limitado
Cumplir con las funcionalidades del programa solicitadas	Si	
Obtener una alta rigidez durante la ejecución	Si	



Documentación interna del programa	Si	.
Documentación externa del programa	Si	
<b>Extra:</b> Creación de un cronometro	Si	
<b>Extra:</b> Sistema de repeticiones		

*Ilustración 13: Objetivos alcanzados*

En el siguiente enlace se puede encontrar el repositorio que contiene el proyecto

<https://github.com/JPHuntV/P3-Lenguajes-S2-2022.git>