



## Proyecto #1

### ESCAPE DEL LABERINTO

Te encuentras atrapado en un antiguo laberinto subterráneo lleno de trampas, tesoros ocultos, obstáculos móviles y portales mágicos que te permiten viajar entre diferentes puntos del mapa. Tu misión es programar un juego en C++ que simula este escape utilizando una secuencia de comandos de movimientos como entrada.

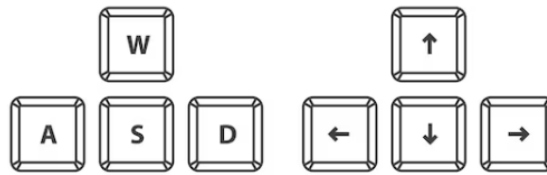
#### Descripción del Laberinto:

- El laberinto está representado por un sistema de coordenadas, cuyos valores siempre **x** y **y** están en el intervalo de 0 a 9 (incluyendo los extremos).
- Cada celda puede contener:
  - **E** (Entrada): el punto de inicio del jugador.
  - **S** (Salida): el objetivo del jugador.
  - **T** (Tesoro): otorga un bonus de vida si lo encuentra.
  - **X** (Trampa): reduce la vida del jugador si cae en ella.
  - **.** (Camino): espacio vacío que permite el paso.
  - **#** (Muro): bloquea el paso del jugador.
  - **P** (Portales): Funcionan en pares. Cada portal está vinculado a otro P, por lo que se asume, creando\* Un enlace bidireccional.

#### Reglas del Juego:

1. **Inicio del Juego:**
  - El jugador comienza con un número específico de puntos de vida, definido en la entrada.
  - El objetivo es llegar a la celda **S** (Salida) antes de que se agote su vida, utilizando la menor cantidad de movimientos posibles y recogiendo el mayor número de tesoros.
2. **Entrada del Programa:**
  - La entrada será por standard input y estará formada de la siguiente manera:
    1. La cantidad de vidas con las que comienza el jugador (**L**).
    2. Límites de las dimensiones del laberinto (**N** y **M**).
    3. La cantidad de elementos en el laberinto.
    4. Lista de cada uno de los elementos de la forma **T X Y** ( **T** el tipo, **X** la posición en el eje **X** y **Y** la posición en el eje **Y**), en el caso de los portales será **P Xa Ya Xb Yb**.
    5. La cantidad de **M** movimientos a ingresar

6. Lista de cada M movimiento posible (w, a, s, d).



Ejemplo gráfico de dirección de movimientos W,A,S,D

### 3. Consideraciones :

- **Muros y Trampas:** Si el jugador intenta moverse hacia un # o los límites, el movimiento no se realiza, y se muestra un mensaje de "Movimiento bloqueado". Si el jugador cae en una X, pierde 10 puntos de vida.
- **Tesoros:** Al encontrar un T, el jugador gana 20 puntos de vida, sin exceder el valor inicial de vidas.
- En el laberinto no existen más de 10 objetos de cada uno de los tipos. Portales, Trampas, Muros, Tesoros.
- La entrada y salida del laberinto son únicas, todo laberinto tendrá siempre una entrada.
- Todos los movimientos ingresados del jugador son de una unidad de distancia.
- Todas las posiciones en el laberinto son números enteros positivos.

### 4. Objetivo

- El jugador debe llegar a la celda S (Salida) antes de que se agote su vida.
- La salida es una evaluación que puede ser:
  - **"LOGRADO"**: Si el jugador llega a la salida con vida.
  - **"ATRAPADO"**: Si el jugador queda dentro en un área del laberinto sin poder salir.
  - **"SORPRENDENTE"**: Si el jugador llega a la salida con vida y ha encontrado todos los tesoros disponibles en el laberinto.
  - **"MUERTO"**: Si la vida del jugador llega a 0 antes de encontrar la salida.

### 5. Salida:

El programa debe mostrar:

- La cantidad de tesoros encontrados: TESOROS: X.
- La cantidad de trampas activadas: TRAMPAS: Y.
- La vida restante del jugador al finalizar: VIDA: Z.
- El estado final del jugador (LOGRADO, ATRAPADO, SORPRENDENTE, MUERTO) .

6. Restricciones:

- No se permite el uso de librerías externas. Solo se pueden usar funciones estándar de C/C++ .
- Se permite el uso de los siguientes temarios para la implementación ([Acciones Elementales](#), [estructuras de control](#) , [Estructuras de Control Interactivo y Funciones](#)).
- Los movimientos se deben procesar uno a uno en el orden en que se encuentran en la cadena de entrada.
- Las salidas del programa son estándar output y deben ser únicas y claras respetando el formato y las indicaciones.

Ejemplo1 de Entrada y Salida:

100 5 5 8 E 1 1 # 1 3 T 2 2 # 2 3 P 2 4 4 3 X 3 1 S 3 5 9 d d d w w a w d d	TESOROS:0 TRAMPAS:1 VIDA:90 LOGRADO
--	--

Condiciones de entrega , restricciones y evaluación:

- Toda lógica, estructuras lógicas, resolución de problema u operación debe ser implementada por el alumno, no se acepta utilización de librerías externas, exceptuando: cstdlib, stdlib, iostream, math.h o csmath.
- Todo el código debe ser entregado en un sólo archivo cpp.
- Debe realizarse un informe donde se explique la implementación, análisis y enfoques utilizados, el mismo tiene ponderación en la evaluación y condiciona la evaluación del proyecto.
- El proyecto se puede realizar sólo con alumnos de la misma sección (mínimo 2 personas y máximo 3).

- El informe y el código debe ser entregado en un zip. con el formato siguiente:  
[AyP] P1-NOMBRE1-APELLIDO1-CEDULA1\_NOMBRE2-APELLIDO2-CEDULA2\_NOMBRE3-APELLIDO3-CEDULA3.zip, cuyo contenido debe ser codigo.cpp y informe.pdf en el correo [aypucv@gmail.com](mailto:aypucv@gmail.com), [thedanidacosta@gmail.com](mailto:thedanidacosta@gmail.com), [ranaldoraffaele@gmail.com](mailto:ranaldoraffaele@gmail.com) y con el asunto replicando el nombre del archivo .zip, de no cumplir el formato pedido su proyecto será sancionado.
- El envío del proyecto debe realizarlo un sólo miembro del equipo, si el mismo es enviado 2 veces, se asumirá como copia.
- La fecha de entrega de proyecto estipulada es: **08/01/2025 11:59pm**, la misma puede estar sujeta a modificación por el **GDAyP**.
- Si al realizar la evaluación del proyecto, el mismo tiene problemas de compilación este no podrá ser evaluado y la nota estará sujeta solo al informe.
- En caso de detección de copia y/o utilización de herramientas de IA, se le asignará la calificación mínima, sin posibilidad de apelación además de sanciones adicionales.

Ejemplo2 de Entrada y Salida:

100 5 5 8 E 1 1 # 1 3 T 2 2 # 2 3 P 2 4 4 3 X 3 1 S 3 5 13 d d d w a s a w d d w d w	TESOROS:1 TRAMPAS:2 VIDA:100 SORPRENDENTE
---	--