Laboratorio 1: Arquitectura y Organización de Computadores

Profesor: Viktor Tapia y Mauricio Solar
Ayudante de cátedra: Diego Acevedo y Claudio Jiménez
Ayudante de Tarea: Vicente Alvear y Luciano Yevenes

19 de Marzo 2025

1 Reglas Generales

Para esta tarea se utilizará python para crear un pequeño juego que implemente conversión de bases númericas que cumpla con los requerimientos estipulados en la sección 2. Deberá incluir un README con la identificación de los estudiantes que desarrollaron la tarea, además de cualquier supuesto utilizado.

2 METAL GEAR SOLID 1010: BINARY SNAKE

ARMando y su primo RISCardo , que crecieron jugando los juegos de sigilo *Metal Gear Solid* han decidido hacer su propia versión del juego. Sin embargo su computadora es muy limitada y requiere que varios elementos del juego sean diseñados usando bases distintas a la decimal para funcionar apropiadamente. Para eso le han encargado a usted, con su vasto conocimiento en la asignatura **INF245 ARQUITECTURA DE COMPUTADORES**, generar la base del juego para que los primos puedan crear la secuela de sus sueños.

2.1 Lógica de tablero

El juego inicia solicitando dos variables de input por consola, el largo del pasillo y la cantidad de guardias. El largo se va a usar para definir las proporciones del pasillo, mientras que la cantidad de guardias define cuantos guardias se deben generar.

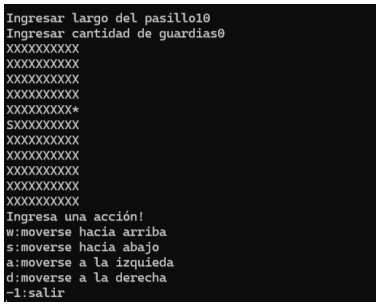

Figure 1: Tablero pequeño

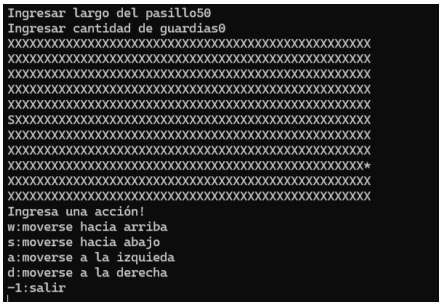

Figure 2: Tablero mediano

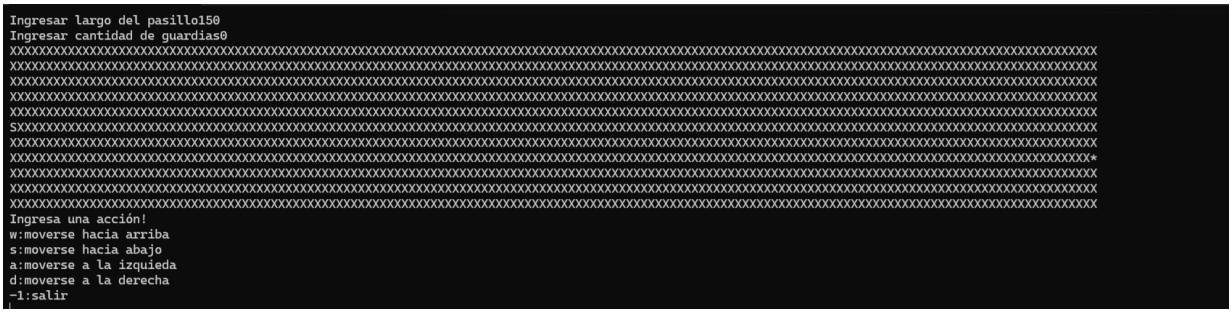

Figure 3: Tablero grande

Las proporciones siempre van a ser x*11 con x siendo el largo ingresado. Esto va a permitir generar una matriz rectangular que facilita el movimiento y la corrección.

2.1.1 Snake

El jugador siempre aparece en la columna 0 en la posicion de al medio [0,5] y es representado con una 'S'.

2.1.2 Objetivo

El objetivo aparece en una ubicación aleatoria dentro de la última columna y es representado con un '*'

2.1.3 Guardias

Los guardias deben generarse aleatoriamente en el tablero según la cantidad ingresada en el input y deben ser representados con un '!'

El resto de casillas se pueden representar con un cáracter general como una 'X' (ver ejemplo)

2.2 Lógica de movimiento (Conversión Bases → Decimal)

El movimiento de snake puede ser en las 4 direcciones cardinales y la base númerica en la cual debe ingresarse va a depender del tamaño del tablero:

- Tableros con un largo menor a 20 deben recibir la distancia de movimiento en binario.
- Tableros con un largo mayor a 20 pero menor a 100 deben recibir la distancia en octal.
- Tableros con un largo mayor a 100 deben recibir la distancia en hexadecimal.

Figure 4: Prompt de movimiento para un tablero pequeño. El input es en la base solicitada y luego el programa debe poder convertirla a decimal.

El programa debe tomar el input y hacer la conversión pertinente a decimal para luego desplazarse en esa dirección. Hay 3 resultados para un movimiento válido:

- Si el jugador se mueve a una celda con un guardia (!) el juego termina.
- Si el jugador se mueve a una celda vacía Snake(S) aparece en esta celda y el ciclo continúa.
- Si el jugador llega al objetivo (*) se inicia la última etapa del juego, el desafío o hackeo.

2.3 Lógica de hackeo (Conversión Decimal → Bases)

La segunda parte del programa interactúa con el usuario de la siguiente forma:

1. El programa genera un número al azar (los rangos dependen nuevamente del tablero)
2. El programa lo convierte a la base del tablero y lo imprime en consola.
3. El usuario debe ingresar en decimal el número impreso.
4. El programa convierte el número ingresado a la base del tablero. Si es equivalente al número aleatorio generado el jugador gana el juego, de lo contrario, lo pierde.

Los rangos son:

- Pequeño : 0-20
- Mediano : 0-100
- Grande : 0-500

3 README

Debe contener como mínimo:

- Nombre, RoI y Paralelo de los integrantes.
- Especificación de los algoritmos y desarrollo realizado.
- Supuestos utilizados

4 Consideraciones

- Se deberá trabajar de a pares.
- Se deberá entregar en Aula a mas tardar el día 10 de abril de 2025 hasta las 23:59 horas. Se descontarán 5 puntos por cada hora o fracción de atraso. Las copias serán evaluadas con nota 0 en el promedio de las tareas.
- La tarea debe realizarse usando el lenguaje python. Se asume que ha tenido vivencias con él, o que aprende con rapidez.
- Pueden crear todas las funciones auxiliares que deseen, siempre y cuando estén debidamente comentadas.
- La entrega considera un único archivo de nombre **t1.py** junto con el README. Los archivos deberán ser comprimidos y enviados en un archivo .tar.gz o en .zip (esto queda a su conveniencia) en el formato **LAB1_ROL1_ROL2.**
- Si no se entrega README, o si su programa no funciona, la nota es 0 hasta la recorrección.
- Una vez entregadas las notas de la tarea existirá un plazo de 5 días para apelar. Transcurrido este plazo las notas no podrán ser modificadas.
- Usted debe implementar la conversión en ambos sentidos. **ESTA EXPLICITAMENTE PROHIBIDO CUALQUIER LIBRERÍA O FUNCIÓN QUE DERROTE EL PROPÓSITO DE LA TAREA (ES DECIR, QUE HAGAN LAS CONVERSIONES DE BASE), SI NO SE CUMPLE ESTO TENDRÁ NOTA 0**