

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Estructuras de Datos
Practica #1
Ingenieros

- Ing. Luis Espino
- Ing. Jesús Guzmán
- Ing. Álvaro Hernández

Auxiliares

- Dennis Masaya
- Ricardo Cutz
- Antonio Hernández



SNAKE RELOADED

Objetivos

1. Familiarizarse con el lenguaje de programación Python
2. Familiarizarse con la Librería Curses de Python
3. Comprender y desarrollar distintas estructuras de datos lineales como lo son listas, listas enlazadas dobles, listas circulares, pilas y colas.
4. Definir algoritmos de búsqueda y recorrido.
5. Familiarizarse con la herramienta Graphviz para la generación de reportes de estructuras graficos.

Definición del problema

Se le solicita al estudiante desarrollar una aplicación de escritorio del famoso juego "SNAKE", esta será utilizada en computadoras que serán donadas a escuelas de bajos recursos en el sector rural de Guatemala, las computadoras cuentan con un procesador y memoria RAM de gama baja por lo que es indispensable que el juego se realice por medio de consola ya que el uso de interfaz grafica podría llevar a que se congelaran estas computadoras.

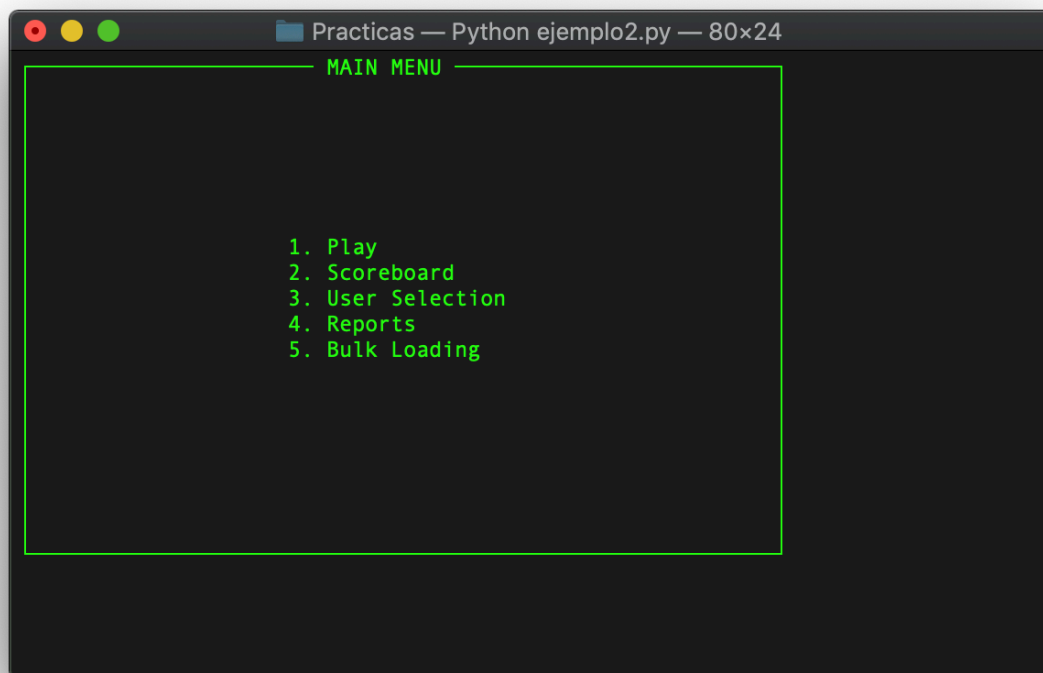
Descripción

Según los requerimientos antes mencionados se deberá de desarrollar el juego “SNAKE RELOADED” el cual no es mas que una modificación del clásico juego de “SNAKE”, este juego consistirá en una serpiente la cual se maneja a través de una pantalla comiendo bocadillos, y creciendo o disminuyendo como consecuencia de dichos bocadillos, se deberá de contar con distintos niveles y un sistema de puntaje y usuarios que le permitirá al usuario tener la mayor diversión posible.

Menú Principal

Al comenzar a correr la aplicación se debe de tener un menú principal con las siguientes opciones:

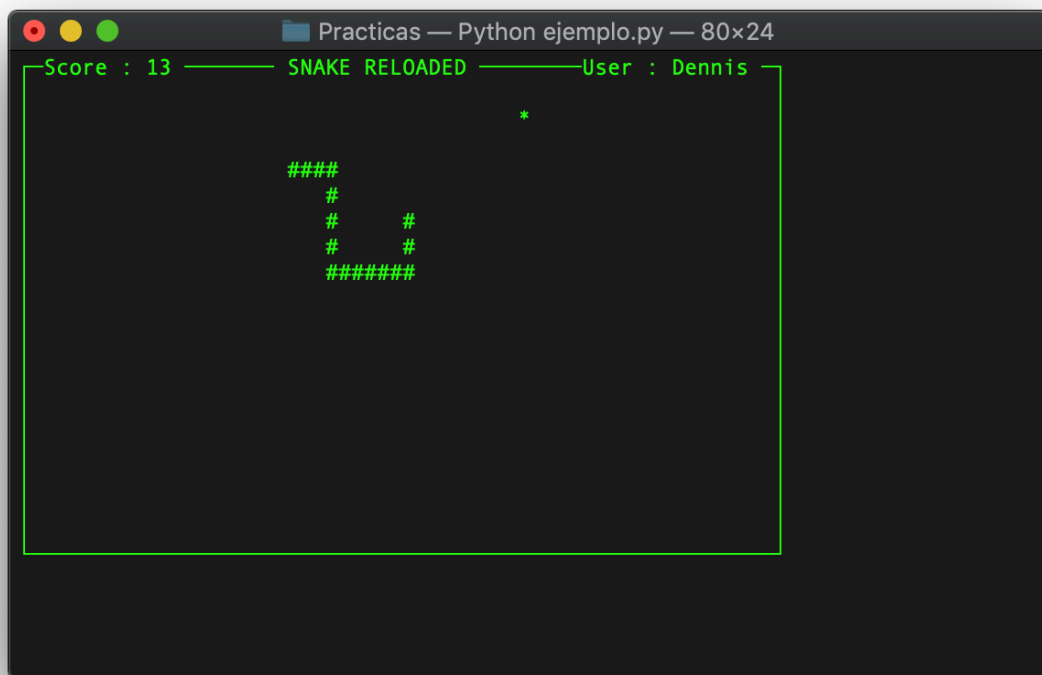
1. *Play (Jugar)*
2. *Scoreboard (Puntuaciones)*
3. *User selection (Usuarios)*
4. *Reports (reportes)*
5. *Bulk loading (carga masiva)*



1. PLAY (JUGAR)

Las reglas del juego son simples, se debe de tener una serpiente con un tamaño inicial de 3, la serpiente se maneja dentro del campo de juego buscando comer bocadillos, los bocadillos pueden ser de 2 tipos (explicados mas adelante), además se deben de manejar 2 o mas niveles según el estudiante lo requiera, se debe de llevar la puntuación actual y el usuario actual con el que se esta jugando impreso en la pantalla. (Nota: si al empezar el juego el usuario aun no ha elegido un Usuario registrado para jugar se le deberá pedir la información para crear uno y se deberá de empezar a jugar con el usuario recién creado.)

La serpiente se debe de manejar mediante una **LISTA ENLAZADA DOBLE**, debido a que debe de haber una tecla asignada que permita que la serpiente cambie de dirección en el momento que dicha tecla sea presionada.



```
Score : 13 ——— SNAKE RELOADED ——— User : Dennis
                                     *
      ####
      #
      # #
      # #
      #####
```

a. SCORE (PUNTUACIÓN)

Se deberá de llevar registro de la puntuación actual por medio de una **PILA**, para avanzar al siguiente nivel se deberá de contar con un mínimo de 15 puntos.

b. LEVELS (NIVELES)

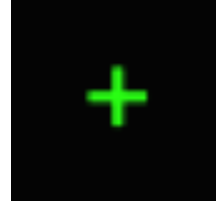
El desarrollo de niveles y su dificultad quedara a discreción del usuario, por lo menos se deberán de manejar 2 niveles, cada nivel deberá de contener una dificultad mayor que el anterior (escenario mas pequeño, mayor velocidad del juego, obstáculos en la pista, etc.).

c. SNACKS (BOCADILLOS)

Se manejaran 2 tipos de bocadillos en el juego (+) y (*)

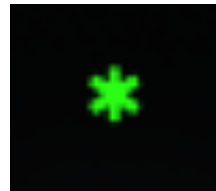
i. SNACK 1 (+)

Este tipo de bocadillo realiza un push() en la pila que se utiliza para llevar el score o puntaje, y hace que la serpiente crezca.



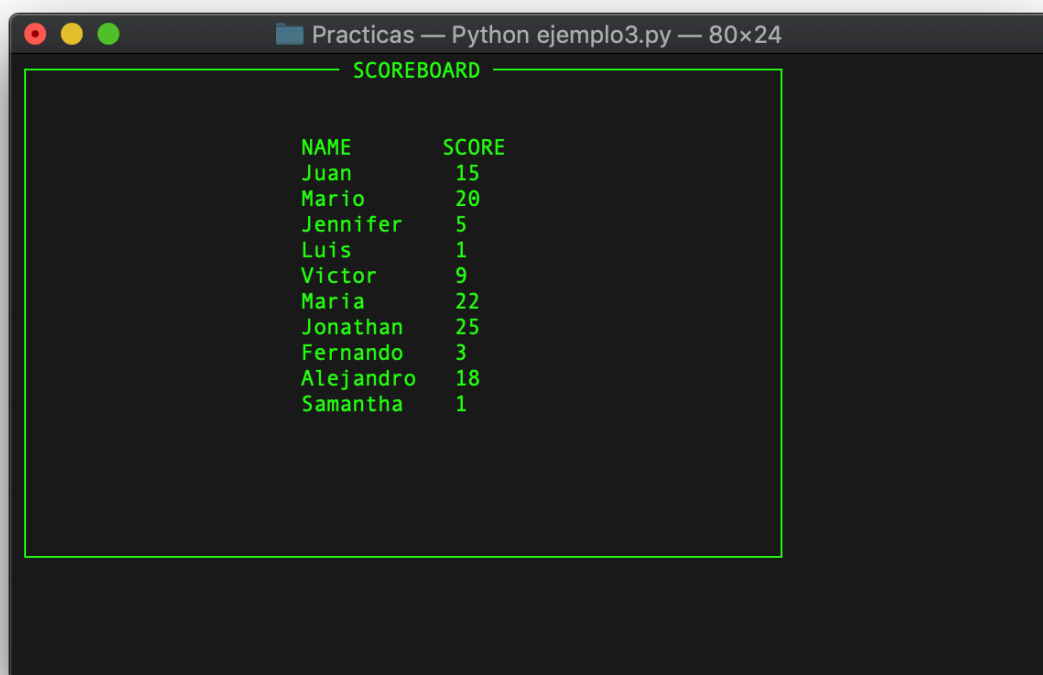
ii. SNACK 2 (*)

Este tipo de bocadillo realiza un pop() en la pila, remueve un punto del score o puntaje, y hace que la serpiente disminuya.



2. SCOREBOARD (PUNTUACIONES)

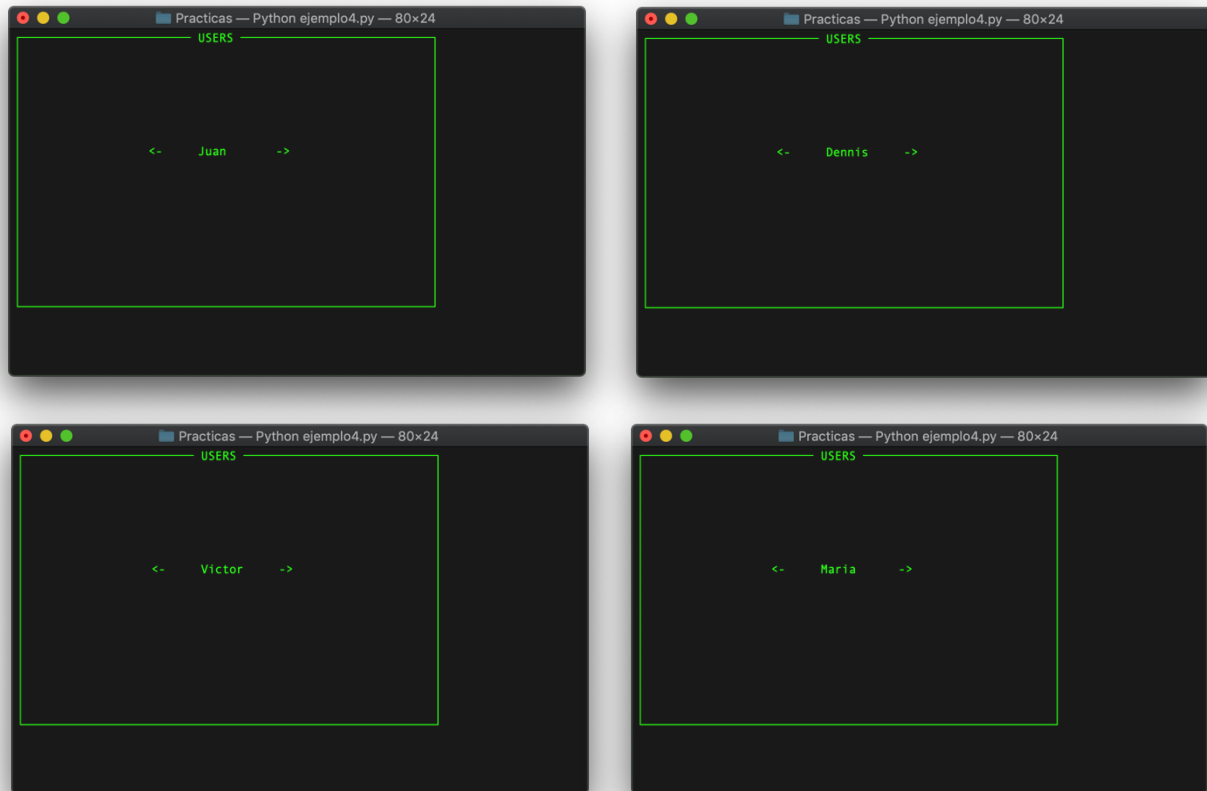
Se debe de manejar una opción de scoreboard, esta opción contendrá una lista de las ultimas 10 puntuaciones que tiene el juego, cada registro de esta lista debe de contener el nombre del usuario que realizo la puntuación y la puntuación, el scoreboard deberá de ser implementado por medio de una **FILA**, tomar en cuenta que la estructura debe de ser dinámica, es decir si únicamente existen 3 puntuaciones, la fila únicamente tendrá un tamaño de 3, una vez alcance un tamaño de 10, se debe de empezar a eliminar datos mediante el método **unqueued()** de la **FILA**.



NAME	SCORE
Juan	15
Mario	20
Jennifer	5
Luis	1
Victor	9
Maria	22
Jonathan	25
Fernando	3
Alejandro	18
Samantha	1

3. USER SELECTION (*USUARIOS*)

Se debe de implementar una ruleta de selección de usuarios, la cantidad de usuarios que se pueden ingresar en la aplicación únicamente esta determinado por la cantidad de memoria disponible en la computadora, si se llega al final de la lista se debe de circular de vuelta al principio, por lo mismo se debe de trabajar por medio de una **LISTA CIRCULAR DOBLEMENTE ENLAZADA**.



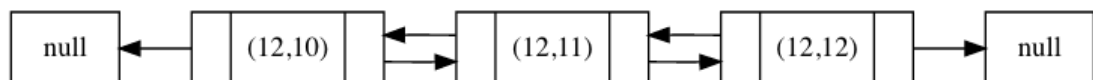
4. REPORTS (*REPORTES*)

Se debe de implementar una sección de reportes, esta sección contendrá un submenú el cual permitirá generar reportes de las estructuras utilizadas en el proyecto, **TODOS** los reportes deberán de ser implementados por medio de la herramienta **GRAPHVIZ**.

a. SNAKE REPORT

Este reporte se deberá generar sobre el estado de la serpiente en el momento de pausa o de "Game Over", se debe de mostrar la lista enlazada doble que representa la serpiente.

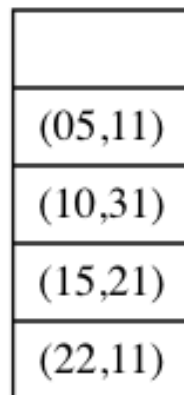
Ejemplo: si la serpiente actualmente es de tamaño 3, el graphviz debería quedar de la siguiente manera (los números entre paréntesis representan las coordenadas (x,y) de cada nodo de la serpiente):



b. SCORE REPORT

este reporte se debe de generar sobre el estado actual del Score, se debe de mostrar de forma clara la pila que se implemento para manejar el score del juego.

Ejemplo: si actualmente tiene un puntaje o score de 4 puntos, se debe de mostrar la pila correspondiente con las coordenadas (x,y) de los bocadillos que representan ese puntaje (también se debe de mostrar un cuadro vacío arriba de la pila representando el tope de la pila):



c. SCOREBOARD REPORT

este reporte se realiza sobre la sección de scoreboard, debe de tener la fila implementada en dicha lista con su información correspondiente.

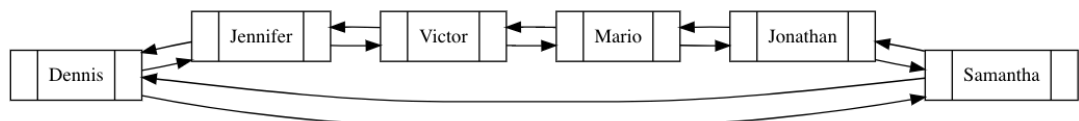
Ejemplo: si actualmente únicamente se han jugado 3 juegos, se debe de mostrar a siguiente FILA que representa dichos juegos:



d. USERS REPORT

este reporte se realiza sobre la sección de User selection, se debe de mostrar la lista circular utilizada para guardar la información de cada usuario.

Ejemplo: si actualmente se tienen registrados 6 usuarios en el sistema, se debería de mostrar la siguiente lista circular doblemente enlazada que representa dichos usuarios:



5. BULK LOADING (CARGA MASIVA)

Se deberá de contar con una opción de carga masiva, la cual permitirá la carga masiva de usuarios, para esto se proporcionará un archivo .CSV con el siguiente formato: (1 sola columna con un encabezado que dice “Usuario”, seguido de todos los usuarios que se desean ingresar.

A14				
	A	B	C	D
1	Usuario			
2	Victor			
3	Mario			
4	Josue			
5	Carlos			
6	Randy			
7	Dennis			
8	Maria			
9	Samantha			
10	Jimena			
11	Juan			
12	Andres			
13	Estuardo			
14	Luis			
15				
16				

Restricciones

Las estructuras deben de ser desarrolladas por los estudiantes sin el uso de ninguna librería o estructura predefinida en el lenguaje a utilizar Los reportes son esenciales para verificar si se trabajaron correctamente las estructuras solicitadas, Si no se tiene el reporte de alguna estructura se anularan los puntos que tengan relación tanto al reporte como a la estructura en cuestión.

Penalizaciones

1. Falta de seguimiento de desarrollo continuo por medio Github tendrá una penalización del 10%.
2. Falta de seguimiento de instrucciones conforme al método de entrega (nombre del repositorio) tendrá una penalización del 5%.
3. Falta de puntualidad conforme a la entrega tendrá una penalización de la siguiente manera:
 - a. 1-10 minutos – 10%.
 - b. 11-59 minutos – 30%.
 - c. Pasados 60 minutos tendrá una nota de 0 y no se calificara.

Observaciones

1. Lenguaje a utilizar: **Python**
2. Herramienta para desarrollo de reportes gráficos: **Graphviz**.
3. La aplicación debe de ser capaz de generar y abrir con un visor de imágenes predeterminado las imágenes generadas con Graphviz.
4. La entrega se realizará por medio de: **Github**, cada estudiante deberá crear un repositorio con el nombre: **EDD_1S2019_P1_#carnet**, y agregar a su auxiliar correspondiente como colaborador del mismo, para poder analizar su progreso y finalmente a partir del mismo repositorio realizar la calificación correspondiente.
5. Además de tener a su auxiliar como colaborador del repositorio para tener un control y orden de las personas que entreguen deberán de colocar el Link de su repositorio en la Tarea que cada auxiliar asignara en su classroom correspondiente.
6. Fecha y hora de entrega: **Miércoles 14 de agosto, a las 23:59 horas**.
7. **Copias serán penalizadas con una nota de 0 y castigadas según lo indique el reglamento.**