**Proyecto Análisis de Algoritmos**


**Presentado por:**
Iván Alejandro Martínez

**Presentado a:**
Cristhiam Campus Julca

**Pontificia Universidad Javeriana**
**Análisis de Algoritmos**
**Bogotá D.C, octubre 7 de 2022**
**Investigación Proyecto**

# Análisis algorítmico juego Snake

Iván Alejandro Martínez Gracia

*Pontifica Universidad Javeriana*

7 de octubre de 2022

**Resumen**

Dentro del desarrollo de software contamos con distintas ramas, algunas de ellas enfocadas a la optimización de recursos como tiempo y hardware, estas ramas siempre intentaran buscar hacer algoritmos más potentes con el menor consumo de recursos posible. Para determinar la eficiencia contamos con bastantes herramientas matemáticas, y de programación como los son los distintos paradigmas y lenguajes. Para poner en práctica estos conocimientos y herramientas, se realizó la búsqueda del código fuente de distintos juegos sencillos, escogiendo a Snake para intentar recrearlo y analizarlo algorítmicamente.

**PALABRAS CLAVE:** software, complejidad, tiempo, programación

## 1. Introducción

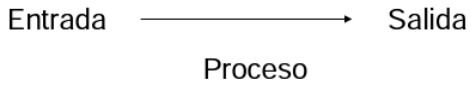
El Snake es un videojuego lanzado en la década de los setenta, bastante popular hasta la época, recordado por ser uno de los primeros juegos de celular. En el 98 estalló masivamente por ser el juego insignia de los celulares Nokia, desde esa primera versión ha sido actualizado y reacondicionado para distintas plataformas, mejorando su aspecto, diseño y jugabilidad. El objetivo del juego consiste en alimentar una serpiente con puntos que van apareciendo aleatoriamente en dentro del cuadrado de la pantalla, la cual estaba delimitada por paredes. Cada vez que la serpiente comiera el punto, esta se extiende un poco más, haciendo más difícil su movilidad por el mapa, además se genera un nuevo punto aleatorio para poder ir por él. Cabe resaltar que la serpiente no se puede chocar con los limites de la pantalla, ni con ella misma, ya que se perdería el juego. El objetivo es poder conseguir la puntación mas alta y que la serpiente sea lo más larga posible.

En principio este es un juego lógico, realizado en su momento con circuitos lógicos, es decir, que es bastante sencillo en su comportamiento. Utilizando herramientas y lenguajes de programación más modernas como Python, con su gran número de librerías, es sencillo poder recrearlo lo más parecido posible al juego original, donde su interfaz gráfica es bastante primitiva y sencilla. Para poder entender todos los aspectos que intervienen en el desarrollo del proyecto se continuara al marco teórico, donde se explica los conocimientos básicos sobre el tema.

## 2. Marco Teórico

Se parte de los conocimientos mas básicos, entendiendo como primer aspecto, ¿Qué es un algoritmo?

**Algoritmo:** Se entiende como una sucesión de pasos bien definidos y precisos para dar solución a un problema dado.




Por pasos se refiere al conjunto de acciones u operaciones que se efectúan con ciertos aspectos.

Un algoritmo debe ser preciso, determinístico y finito. Además, debe ser general, es decir, que sea capaz de resolver una clase de problemas lo mas amplia posible. Debe ser eficiente, en cuantos menos recursos en tiempo, espacio y procesamiento consuma mejor.

**Complejidad Algorítmica:** *"La complejidad algorítmica representa la cantidad de recursos (temporales) que necesita un algoritmo para*

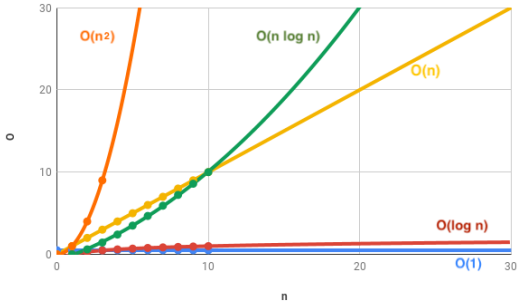
*resolver un problema y por tanto permite determinar la eficiencia de dicho algoritmo."* [1]

El tiempo obtenido debe ser igual al número de pasos o de operaciones simples que e ejecuten para dar solución al problema. En la gran mayoría de los casos el costo depende del tamaño de los datos. Al momento de evaluar el costo se consideran tres posibles casos:

- El caso promedio Θ
- El mejor caso Ω
- El peor caso O

La notación asintótica esta señalada con las letras griegas theta para el caso promedio, omega para el mejor caso y ómicron para el peor de los casos.

Para saber el comportamiento de un algoritmo el caso con mayor relevancia siempre va a ser ómicron, es decir, el peor de los casos, ya que nos va a describir el comportamiento en la peor situación a la que se puede enfrentar el programa. Para hallar el peor de los casos se debe tomar el camino en el que se enfrente cada situación y poder recorrer todo el algoritmo. Se debe obtener una función que describa el tiempo que tarda con n numero de datos. En síntesis, la función debe



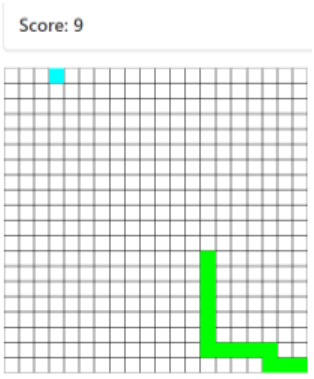

tener como x el número de datos, y como *y* el tiempo que tardo con esa cantidad de datos.

El tipo de función esta dado por la construcción del algoritmo, por ejemplo: cuando se anidan ciclos aumenta el grado de la función, es decir, su exponente, haciendo mas complejo el algoritmo. Como se sabe hay distintas maneras de programar y dar solución a los problemas, pero así mismo habrá algunas soluciones más eficientes, mostrando que su curva esta lo mas pegada al eje x.

- O (1) Constante
- O (n) Lineal
- O ($n^2$) Cuadrática
- O ($n^a$) Polinomial
- O ($\log n$) Logarítmica
- O ($2^n$) Exponencial

## 3. Antecedentes

*"En el juego, el jugador o el usuario controla un elemento alargado y estrecho que se parece a una serpiente, que gira alrededor de un plano delimitado, reconociendo los alimentos, tratando de evitar golpear su propia cola o las "paredes" que rodean el área de juego. Aparte de eso, hay un concepto de comida. Cada vez que la cabeza de la serpiente choca con el objetivo, su longitud sube un poco más y gana un punto. Cuanto más larga sea el gusano, más difícil sería controlar y manejarlo. Es decir, aumenta la dificultad del juego. El usuario controla la dirección de la cabeza del gusano con los botones de direcciones (arriba, abajo, izquierda o derecha) y el cuerpo de la serpiente la sigue."* [2]

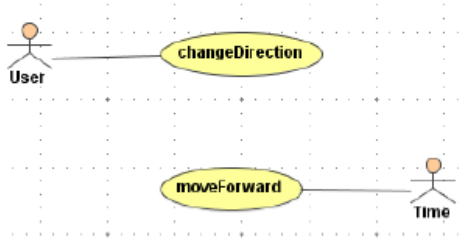



[2]

*"Cada vez que consigue dar a la comida, se le suma una celda más y gana un punto.*

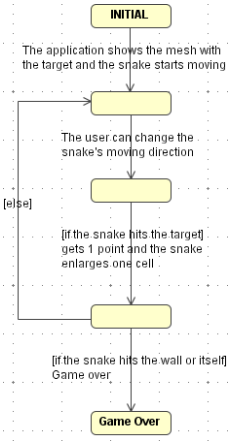
*Para implementar la animación, se ha utilizado el método, setTimeout. En cada iteración con cambios de asignación del espacio se consigue presentar por la pantalla el movimiento. Es cuestión de asignarle a las coordenadas, el color preciso y eso traduciría a implementar el movimiento."* [2]

**Caso de uso:**



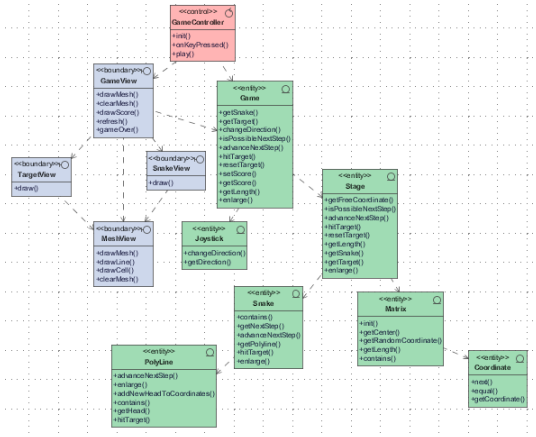

[2]

**Diagrama de estados:**




[2]

**Implementación:**



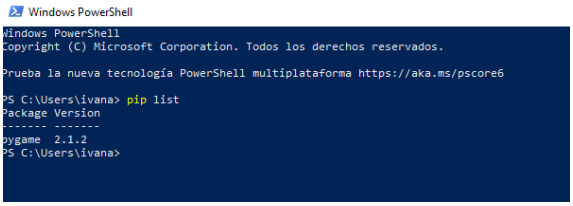
[2]

## 4. Propuesta

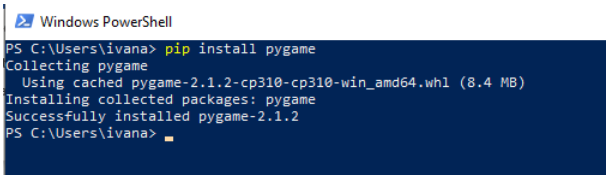
Teniendo en cuenta los distintos lenguajes de programación que existen, se opta por recrear el juego en Python, ya que es uno de los más versátiles y con mas herramientas que facilitan el desarrollo del juego.

Python cuenta con la librería pygame "*la cual **es** una librería para el desarrollo de videojuegos en segunda dimensión 2D. Pygame está basada en SDL, que **es** una librería que nos provee acceso de bajo nivel al audio, teclado, ratón y al hardware gráfico de nuestro ordenador.*" [3]

Para el buen desarrollo de este juego y ejecución se deberá tener instala la librería pygame. Para verificar la existencia de esta librería en el equipo se tiene que ejecutar el comando `pip list`, el cual mostrara las librerías de Python instaladas.




En caso de no tener la librería instalada se tendrá que ejecutar el siguiente comando desde el shell: `pip install pygame`, tal como se muestra en la imagen:




Se tiene claro el objetivo del juego y su funcionamiento, para ello es necesario poder desplazar una serpiente alrededor del mapa, en busca de frutas para comer, crecer y sumar puntaje. Cada vez que la serpiente coma una fruta crecerá un poco, haciendo más difícil su movilidad. La serpiente no podrá estrellarse contra si misma o hacia un limite del mapa o perderá.

La disposición de las frutas debe ser de manera aleatoria por cada vez que se consuma una. El movimiento de la serpiente solo podrá ser estrictamente en alguna de las cuatro direcciones, ya sea hacia arriba, abajo, hacia la izquierda o derecha. El direccionamiento de la serpiente se hará bajo las flechas del teclado.

La interfaz gráfica del juego será bastante sencilla, el movimiento de la serpiente comenzara en el momento que se empiece a ejecutar el juego, se resaltará a la serpiente y a la futa por encima del mapa, para tener el mayor contraste de movimiento y resaltar el objetivo. En la parte de

arriba del mapa se tendrá un contador de ´puntaje el cual ira avanzando a medida de que la serpiente consuma frutas.

El mapa en el que se desarrollara el juego será una matriz en la que cada casilla representara un píxel, siendo una línea de pixeles iluminados la serpiente, y un píxel unitario la fruta que hay que consumir. En caso de que la serpiente choque con si misma o contra algún límite del mapa aparecerá un mensaje de game over con la puntuación alcanzada.

## 5. Resultados


## 6. Conclusiones

## 7. Bibliografía

[1] D. d. I. U. d. Valladolid, «Universidad de Valladolid,» [En línea]. Available: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www2.infor.uva.es/~jvalvarez/docencia/tema5.pdf. [Último acceso: 5 10 2022].

[2] M. Mokhber, «Trabajo Fin Master,» Julio 2019. [En línea]. Available: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://oa.upm.es/58698/1/TESIS_MASTER_MONA_MOKHBER.pdf. [Último acceso: 2022 10 5].

[3] «Keep Coding Tech School,» [En línea]. Available: https://keepcoding.io/blog/que-es-pygame/. [Último acceso: 5 10 2022].