Universidad Nacional Autónoma de México

Facultad de Ingeniería

Fundamentos de programación

Juego arcade: la serpiente.

Alumna: Rivera González Frida Alison

M.I. Marco Antonio Martínez Quintana

Semestre: 2021-1

Fecha de realización: enero de 2021

Resumen de contenido.

En el avance de este documento se tratarán los temas relacionados a la elección de proyecto y su desarrollo.

Se explicará el significado e historia de los juegos arcade, en donde entra o se categoriza el proyecto, así como las características que este debe de tener para entrar en esta categoría. Se mostrará su desarrollo desde el planteamiento del proyecto, hasta como diferentes actividades han llevado a la realización con éxito de este y su documentación.

Se mostrarán el pseudocódigo, algoritmo, código comentado, diagramas de flujo y de Gantt, así como tablas comparativas respecto a tiempos de realización, materiales, costos y aplicaciones del proyecto.

Para finalizar, el documento incluirá capturas de pantalla del proyecto, en este caso del juego, en funcionamiento, así como el link del canal de YouTube, a donde se subirá la explicación y demostración del videojuego, así como el enlace al repositorio de GitHub, donde se encontrará el código final y una pequeña guía de uso del videojuego.

Juegos arcade.

Introducción.

¿Qué son los juegos Arcade?

Se denomina videojuego Arcade a todos aquellos juegos que están diseñados siguiendo las bases de los antiguos juegos de las máquinas Arcade.

Características de los videojuegos arcade

Para que un juego sea considerado como videojuego arcade debe cumplir con una serie de características:

- Dificultad ascendente.
- Puntaje como objetivo.
- Dificultad ascendente.
- Sistema de vidas.
- Diseño minimalista.
- Pocas interrupciones.

Juego snake

El Snake (a veces también llamado la serpiente) es un videojuego lanzado a mediados de la década de 1970.

En el juego, el jugador o usuario controla una larga y delgada criatura, semejante a una serpiente, que vaga alrededor de un plano delimitado, recogiendo alimentos (o algún otro elemento), tratando de evitar golpear a su propia cola o las "paredes" que rodean el área de juego. Cada vez que la serpiente se come un pedazo de comida, la cola crece más, provocando que aumente la dificultad del juego. El usuario controla la dirección de la cabeza de la serpiente (arriba, abajo, izquierda o derecha) y el cuerpo de la serpiente la sigue. Además, el jugador no puede detener el movimiento de la serpiente, mientras que el juego está en marcha.

Enfoque de emprendimiento.

Este proyecto no es uno muy grande, pero puede ser una opción de emprendimiento. Los juegos arcade tuvieron un gran auge en los años setenta y con el avance y desarrollo de la tecnología que compone diferentes aspectos, estos juegos han quedado poco a poco en el olvido, pero como lo podemos ver, por ejemplo, con la moda, lo clásico siempre vuelve y justamente con los avances tecnológicos, podemos traer viejos juegos a la era moderna.

Con respecto a este juego, se podía encontrar en máquinas portátiles, algunas veces no tan pequeñas, pero su representación visual no era muy buena, los pixeles que utilizaban no eran de la resolución que pueden ser ahora.

En base a esto un enfoque de emprendimiento para este juego y otros más, es traer los juegos arcade al mundo nuevo, en donde las consolas o máquinas portátiles sean más pequeñas, estilizadas, minimalistas o diseñadas al gusto del cliente y que esta tenga la opción de poder jugar la versión clásica o una opción más moderna, en donde los gráficos a puedan ser en tercera dimensión.

Algoritmo:

1. Se abre la ventana del juego.
2. Aparece la serpiente.
3. Comienza a aparecer comida.
   3.1 La comida aparece de forma aleatoria.
4. Se mueve la serpiente a los lugares donde aparezca la comida.
   4.1 Cada vez que se “coma” una manzana, la puntuación aumentará con un punto.
5. Se termina el juego cuando:
   5.1 La serpiente sale de la ventana.
   5.2 Desaparece la ventana del juego.
   5.3 Aparece el mensaje “Perdiste”.

Diagrama de flujo:

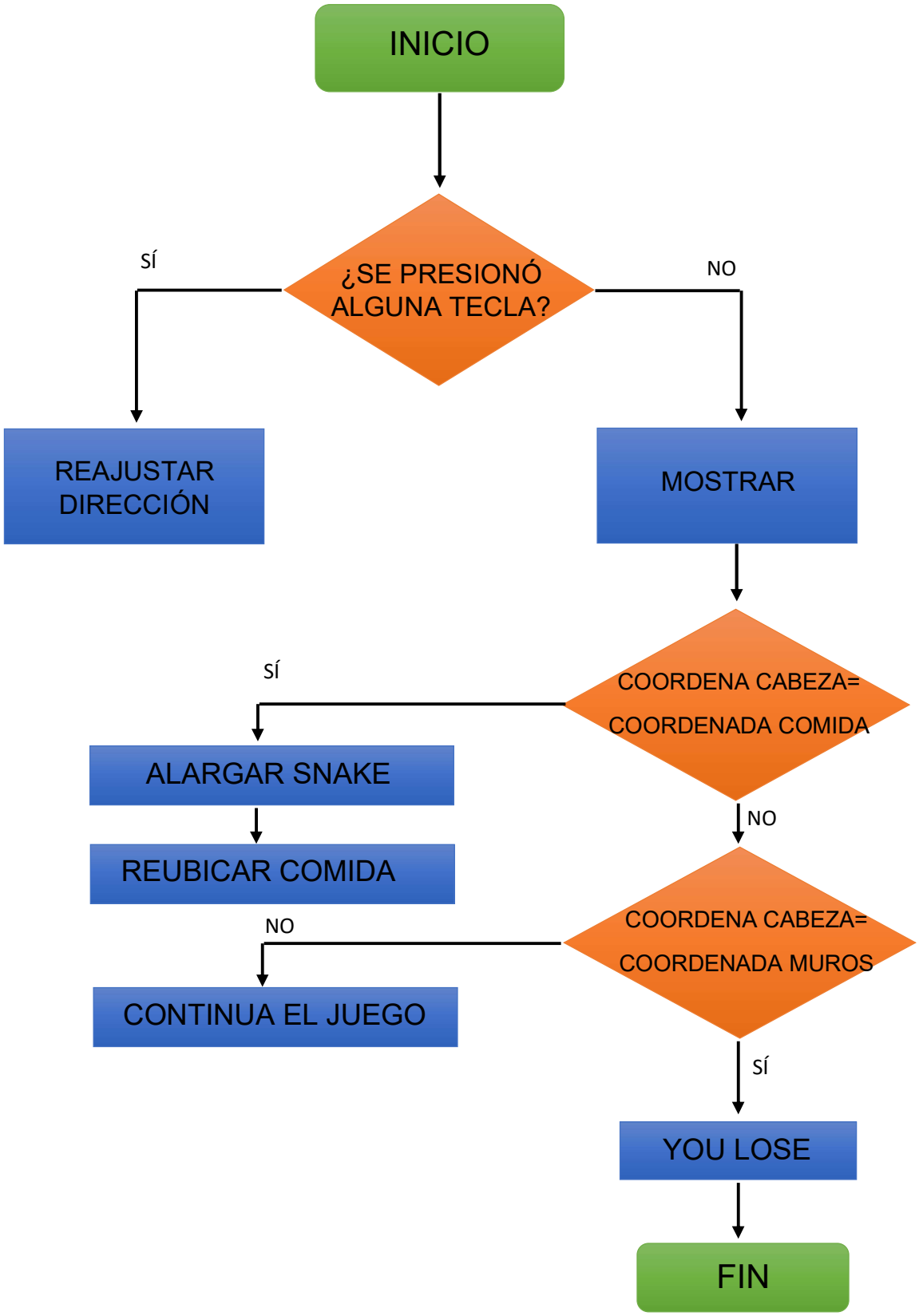

Diagrama de Flujo 1. Juego snake.

Pseudocódigo:

```
Pantalla de carga (load_Screen):
import Tkinter as tk # Python 2
import tkinter as tk # Python 3
import Imagenes #Libreria especial Creada por mi para la carga de Imágenes
import time #Libreria para el manejo de tiempos y cronómetros
class load_window():
   root=object
   def __init__(self): #Declaro el constructor de Clase
      self.root = tk.Tk() #Creo la ventana Tk
      # The image must be stored to Tk or it will be garbage collected.
      imagen=Imagenes.get_imagen("logo2.png",1000,1000) #Llamo la funcion
get_imagen para cargar una imagen a python
      label = tk.Label(self.root, image=imagen,bg="black") #creo un label, el cual
contendrá la imagen
      self.root.geometry("+470+200") #establesco la posición de la pantalla de carga
en el monitor
      self.root.overrideredirect(True) #Le establezco la propiedad de una ventana
traslucida
      self.root.lift() #Le digo que al cargarse deberá aparecer arriba de todas las
ventanas creadas
      self.root.wm_attributes("-topmost", True) #le establezco que no deberá poseer
los botones de minimizar, maximizar y cierre
      #self.root.wm_attributes("-disabled", True) #Si no estuviera comentada y no
existiese la línea anterior deshabilitaría las funciones de los botones antes
comentados
      self.root.wm_attributes("-transparentcolor", "black") #Le digo que la ventana
deberá actuar con transparencia en todo tono negro
      label.pack() #cargo el label

      # Establezco eventos para el ratón
      self.root.bind("", lambda e:self.exit()) #Si el mouse deja la ventana traslucida
      self.root.bind("",lambda e:self.exit()) #o le doy click sobre la imagen
      #debera ejecutar la funcion exit
      self.root.mainloop()
   def exit(self):
      print "entro"
      time.sleep(5) #espera 5s
      self.root.destroy() #cierra esta ventana
      import Snake #importa la librería Snake correspondiente al archivo numero 2
c=load_window() #Creo un objeto que a si vez crea la ventana de carga
```

Pseudocódigo 1. Juego snake.

Código fuente:

```
import pygame, sys, time, random
pygame.init()

play_surface = pygame.display.set_mode((500,500)) #variable que va a contene
r la pantalla del videojuego
font = pygame.font.Font(None, 30) #ponemos tipo de letra
fps = pygame.time.Clock()

def food():#definimos para hacer la comida
    random_pos = random.randint(0,49)*10
    food_pos = [random_pos,random_pos]
    return food_pos #nos devuelve dos números aleatorios de la multiplicació
n donde se hubicará la comida


def main():
    snake_pos = [100, 50] #cabeza de la serpiente
    snake_body = [[100,50],[90,50],[80,50]] #cuerpo de la serpiente
    change = "RIGHT"
    run = True
    food_pos = food() #va a valer lo que vale food_pos, los dos números
    score = 0

    while run:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                run = False
            if event.type == pygame.KEYDOWN: #haremos que la serpiente se mu
eva
                if event.key == pygame.K_RIGHT:
                    change = "RIGHT" #hace que se mueva en dirección a la de
recha
                if event.key == pygame.K_LEFT:
                    change = "LEFT" #dirección izquierda
                if event.key == pygame.K_UP:
                    change = "UP" #dirección hacia arriba
                if event.key == pygame.K_DOWN:
                    change = "DOWN" #dirección hacia abajo
            if change == "RIGHT":
                snake_pos[0] += 10 #efecto de que se mueve
            if change == "LEFT":
                snake_pos[0] -= 10
            if change == "UP":
                snake_pos[1] -= 10 #resta porque nos acercamos a la y
```

```
            if change == "DOWN":
                snake_pos[1] += 10 #sumas porque nos alejamos de y

            snake_body.insert(0,list(snake_pos)) #añadimos al cuerpo lo que
hemos sumado o restado al cuerpo
            if snake_pos == food_pos: #la cabeza de la serpiente está en la
misma posición que la de la comida
                food_pos = food() #si esto sucede la comida se irá a otro si
tio
                score += 1
                print(score)
            else:
                snake_body.pop()

            play_surface.fill((0,0,0)) #pintamos la pantalla

            for pos in snake_body:#creamos el cuerpo de la serpiente
                pygame.draw.rect(play_surface,(200,200,200),pygame.Rect(pos[
0],pos[1],10,10))

            pygame.draw.rect(play_surface,(169,6,6),pygame.Rect(food_pos[0],
food_pos[1],10,10)) #dibujamos la manzana
            text = font.render(str(score),0,(200,60,80)) #marcamos la puntua
ción
            play_surface.blit(text,(480,20))

            if score < 10: #con una puntuación menor a 10 va lento
                fps.tick(10)
            if score >= 10: #con una puntuación mayor o igual la velocidad a
umenta
                fps.tick(20)

            if snake_pos[0] <= 0 or snake_pos[0] >= 500:
                run = False
                print("YOU LOSE")
            if snake_pos[1] <= 0 or snake_pos[1] >= 500:
                run = False
                print("YOU LOSE")

            pygame.display.flip()


main()

pygame.quit()
```

Código 1. Juego snake.

Resultados

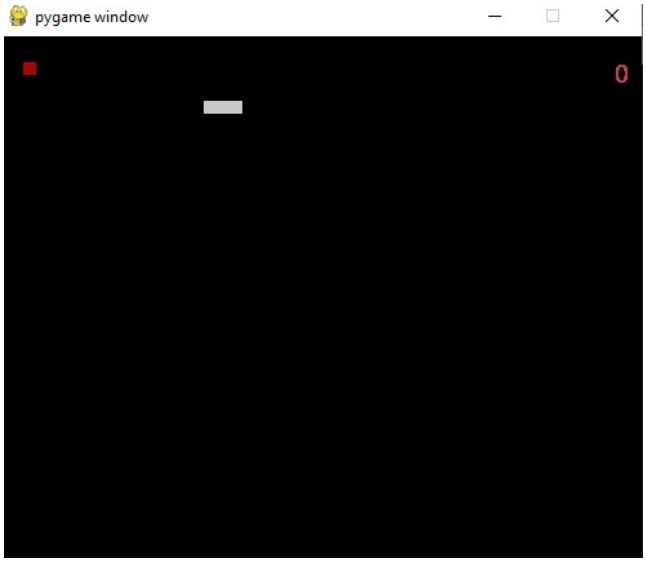

Fig. 1 Inicio del juego snake.

Fig. 2 Desarrollo del juego snake.

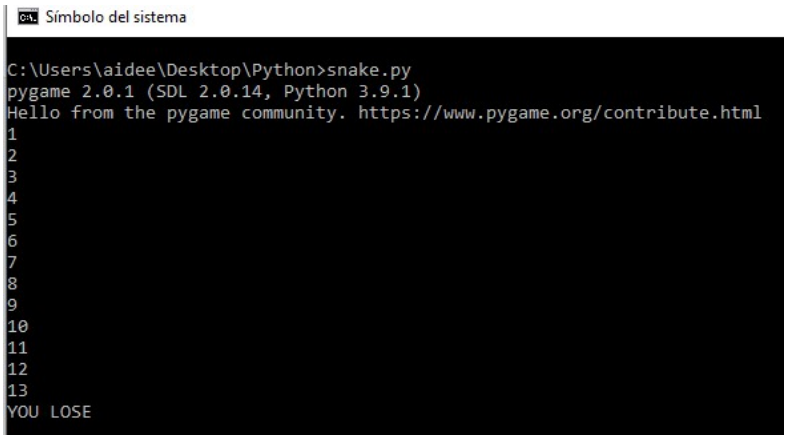

Fig. 3 Puntuación y mensaje de final de juego.

Tabla comparativa de recursos informáticos.

Octubre 2020	Enero 2021
– Computadora	– Computadora
– Investigación de los juegos arcade	– Investigación de los juegos arcade
– Instalación de Python	– Instalación de Python
– Aprender a programar Python	– Instalación de codificador. – (visual studio code)
– Conocer las bases del videojuego	– Aprender programación de Python
– Implementar una buena arquitectura de software	– Conocimiento de las bases de videojuego
	– Arquitectura de software
	– Conocer los colores en Python.
	– Instalar librería Pygame

Tabla 1. Tabla comparativa de recursos informáticos.

Costos asociados:

Octubre 2020	Enero 2021
0$	0$

Tabla 2. Tabla comparativa de costos asociados.

Diagramas de Gantt

- **Octubre – noviembre 2020**

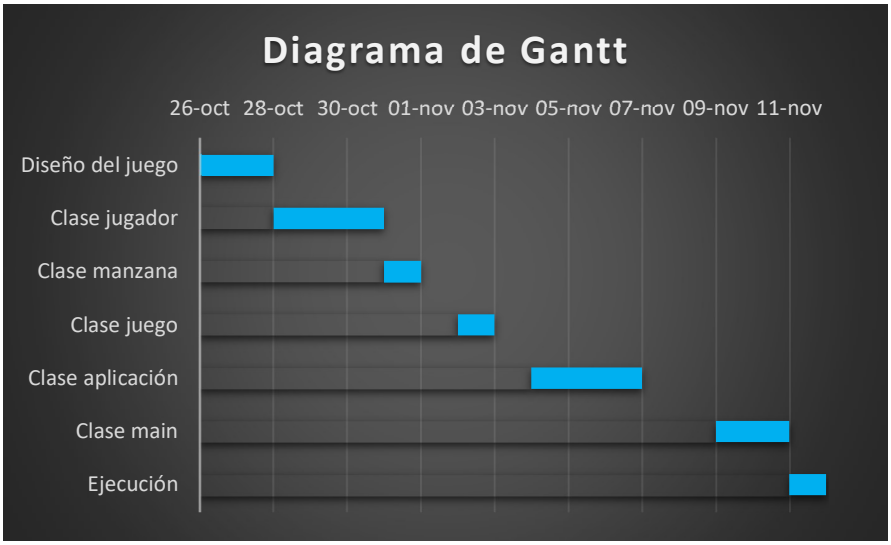

Diagrama de Gantt 1. Octubre-noviembre 2020.

Nombre actividad	Fecha de inicio	Duración en días	Fecha fin
Diseño del juego	26-oct	2	28-oct
Clase jugador	28-oct	3	31-oct
Clase manzana	31-oct	1	01-nov
Clase juego	02-nov	1	03-nov
Clase aplicación	04-nov	3	07-nov
Clase main	09-nov	2	11-nov
Ejecución	11-nov	1	12-nov

Tabla 3. Tabla en relación al diagrama de Gantt, octubre-noviembre 2020.

- **Enero 2021**

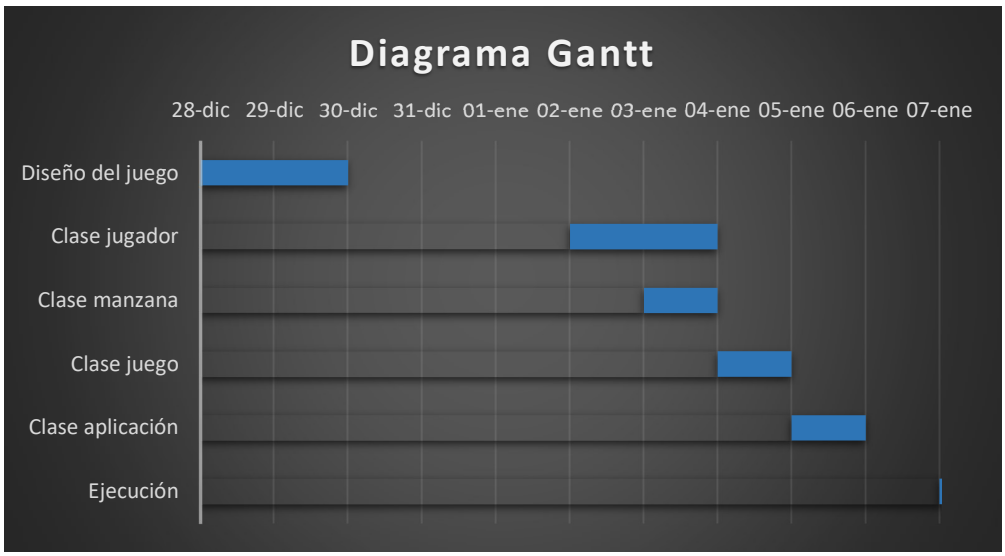

Diagrama de Gantt 2. enero 2021.

Nombre actividad	Fecha de inicio	Duración en días	Fecha fin
Diseño del juego	28-dic	2	29-dic
Clase jugador	02-ene	2	03-ene
Clase manzana	03-ene	1	03-ene
Clase juego	04-ene	1	04-ene
Clase aplicación	05-ene	1	05-ene
Ejecución	07-ene	1	07-ene

Tabla 4. Tabla en relación al diagrama de Gantt, enero 2021.

Canal de YouTube:

https://www.youtube.com/channel/UCJsj2--BFNmXDdiQhgfyhFw/featured

Repositorio GitHub:

Rivera-Frida/Proyecto-Final (github.com)

Guía rápida de usuario:

Para poder utilizar el juego, tenemos que tener Python instalado, así como sugerencia el editor de texto Visual Studio Code,

- Abrimos, desde el computador en donde se quiera jugar, símbolo del sistema, compilamos y corremos el programa.
- Se abrirá la ventana, el juego empieza.
- La comida aparecerá aleatoriamente por la pantalla, para llegar a ella e ir subiendo de puntuación, manejaremos a la serpiente con las flechas en el teclado, para movernos en la dirección querida.
- Se termina la partida cuando la serpiente sale de la ventana y aparecerá tu puntuación junto con el mensaje “you lose”.
- Para volver a jugar, vuelves a correr el programa desde el símbolo del sistema.

Conclusiones:

Python utilizaba una licencia en la que incluía una cláusula, estipulando que la licencia estaba gobernada por el estado de Virginia, por lo que para los abogados de Free Software Foundation (FSF), se hacía incompatible con GNU GLP, así que para que fuera compatible, en las nuevas licencias ya se hicieron las licencias de Python compatibles con GLP, renombrando como Python Software Foundation License. Haciendo que este se convirtiera en un sistema operativo de software libre, en donde los usuarios tengan la libertad de ejecutar, editar, contribuir y compartir el software.

En un inicio el final de este proyecto se veía lejano, el día de hoy está terminado, ha sido un logro personal la culminación de este proyecto. Me gustó, al realizarlo e ir investigando sobre cómo hacerlo o las características de los juegos arcade, han traído la nostalgia de recordar que estos, yo los jugué de niña, recordar como poco a poco fueron quedaron en el olvido, gracias a juegos mucho más desafiantes, con mejor resolución, etcétera, pero yo opino que siempre lo clásico vuelve y que mejor que nuevas generaciones, así a lo mejor ya no haya tantas máquinas arcade, puedan seguir jugando y disfrutando de estos juegos, que muchos pensaran que son sencillos, pero todo lo sencillo tiene su complejidad.

Bibliografía:

- Programando el juego Snake con Python. (2021). SySCursos. Fecha de publicación 9 de enero de 2021. Fecha de consulta enero de 2021. (259) Programando el juego Snake con Python. (2021) - YouTube
- Curso Python para Principiantes. Fazt. Fecha de publicación 22 de enero de 2021. Fecha de consulta enero 2021. (259) Curso Python para Principiantes - YouTube
- "Python" [en línea]. En: Python. (enero 2021). Python - Wikipedia, la enciclopedia libre
- "Arcade juegos" [en línea]. En: género de videojuegos, Glosario. Arcade (género) | Wikijuegos | Fandom

Anexo:

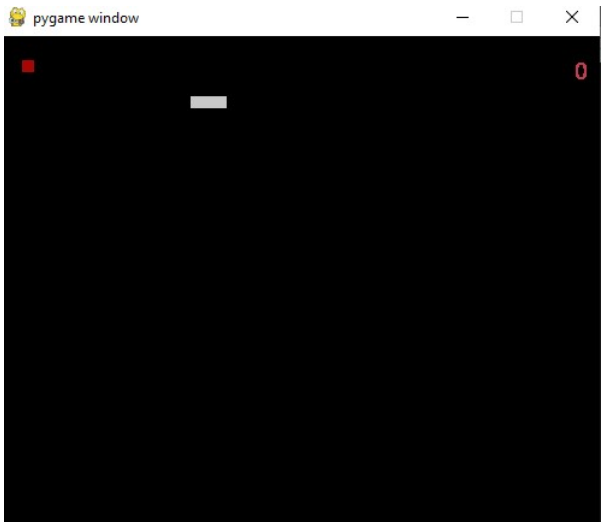

Fig. 1 Inicio del juego snake.

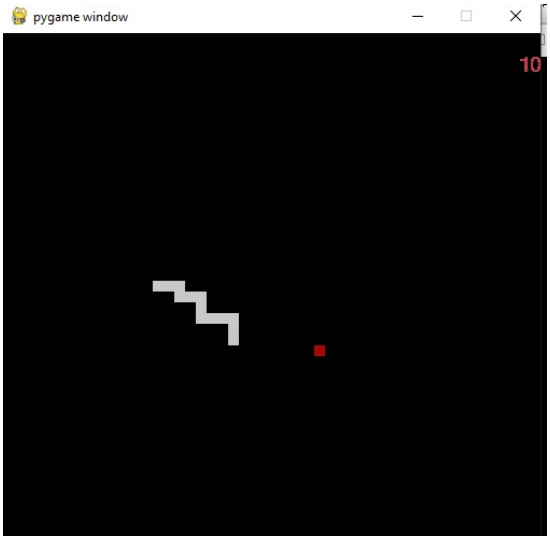

Fig. 2 Desarrollo del juego snake.

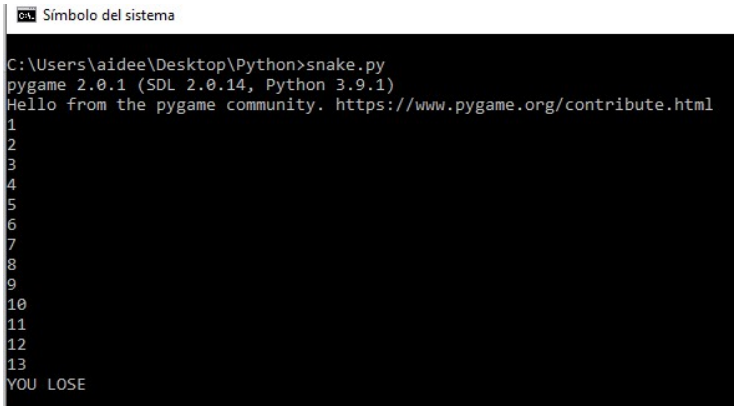

Fig. 3 Puntuación y mensaje de final de juego

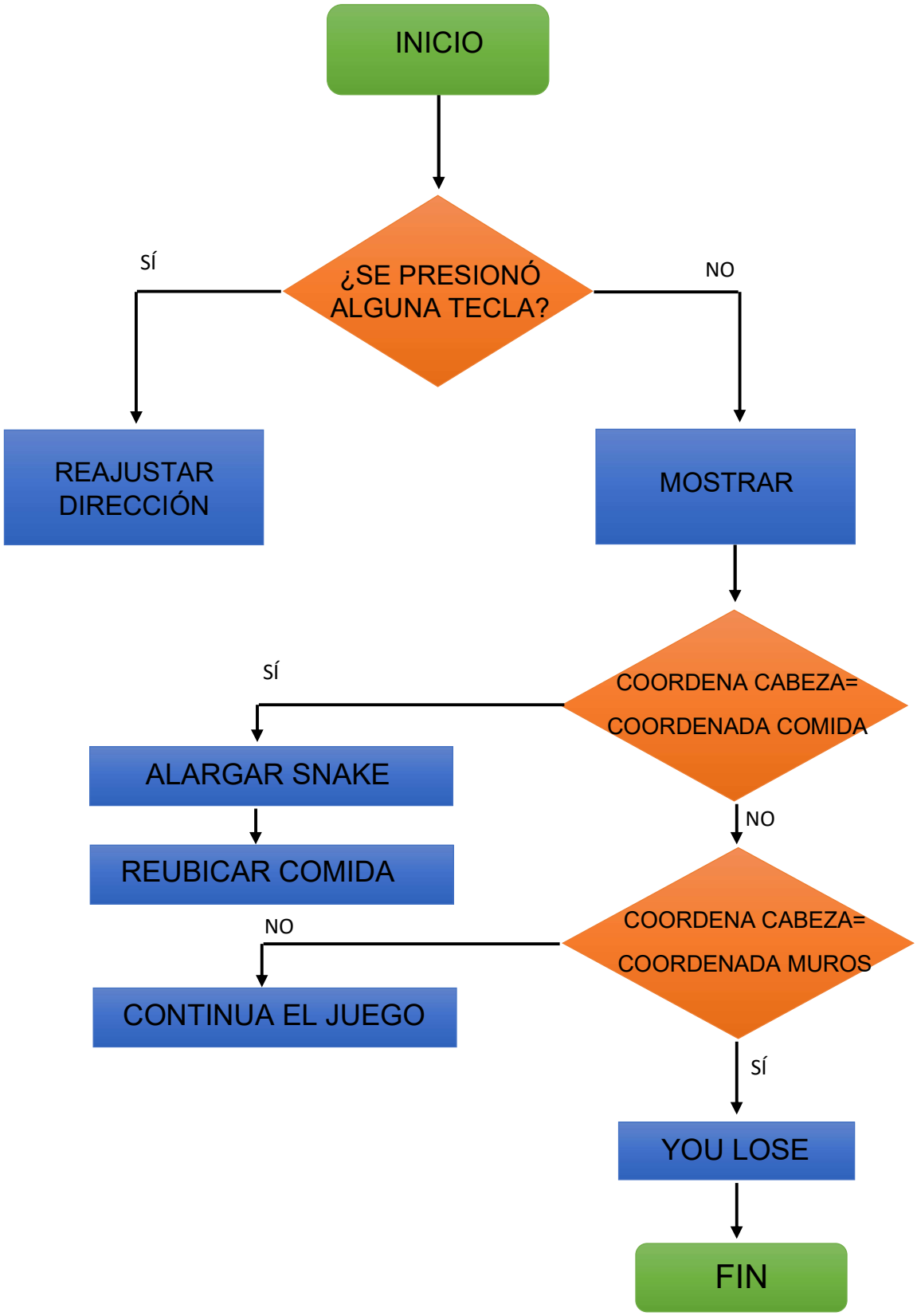

Diagrama de Flujo 1. Juego snake.

Pantalla de carga (load_Screen):

```
import Tkinter as tk # Python 2
import tkinter as tk # Python 3
import Imagenes #Libreria especial Creada por mi para la carga de Imágenes
import time #Libreria para el manejo de tiempos y cronómetros
class load_window():
    root=object
    def __init__(self): #Declaro el constructor de Clase
        self.root = tk.Tk() #Creo la ventana Tk
        # The image must be stored to Tk or it will be garbage collected.
        imagen=Imagenes.get_imagen("logo2.png",1000,1000) #Llamo la funcion
get_imagen para cargar una imagen a python
        label = tk.Label(self.root, image=imagen,bg="black") #creo un label, el cual
contendrá la imagen
        self.root.geometry("+470+200") #establesco la posición de la pantalla de carga
en el monitor
        self.root.overrideredirect(True) #Le establezco la propiedad de una ventana
traslucida
        self.root.lift() #Le digo que al cargarse deberá aparecer arriba de todas las
ventanas creadas
        self.root.wm_attributes("-topmost", True) #le establezco que no deberá poseer
los botones de minimizar, maximizar y cierre
        #self.root.wm_attributes("-disabled", True) #Si no estuviera comentada y no
existiese la línea anterior deshabilitaría las funciones de los botones antes
comentados
        self.root.wm_attributes("-transparentcolor", "black") #Le digo que la ventana
deberá actuar con transparencia en todo tono negro
        label.pack() #cargo el label

        # Establezco eventos para el ratón
        self.root.bind("", lambda e:self.exit()) #Si el mouse deja la ventana traslucida
        self.root.bind("",lambda e:self.exit()) #o le doy click sobre la imagen
        #debera ejecutar la funcion exit
        self.root.mainloop()
    def exit(self):
        print "entro"
        time.sleep(5) #espera 5s
        self.root.destroy() #cierra esta ventana
        import Snake #importa la librería Snake correspondiente al archivo numero 2
c=load_window() #Creo un objeto que a si vez crea la ventana de carga
```

Pseudocódigo 1. Juego snake.

```
import pygame, sys, time, random
pygame.init()

play_surface = pygame.display.set_mode((500,500)) #variable que va a contene
r la pantalla del videojuego
font = pygame.font.Font(None, 30) #ponemos tipo de letra
fps = pygame.time.Clock()

def food():#definimos para hacer la comida
    random_pos = random.randint(0,49)*10
    food_pos = [random_pos,random_pos]
    return food_pos #nos devuelve dos números aleatorios de la multiplicació
n donde se hubicará la comida


def main():
    snake_pos = [100, 50] #cabeza de la serpiente
    snake_body = [[100,50],[90,50],[80,50]] #cuerpo de la serpiente
    change = "RIGHT"
    run = True
    food_pos = food() #va a valer lo que vale food_pos, los dos números
    score = 0

    while run:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                run = False
            if event.type == pygame.KEYDOWN: #haremos que la serpiente se mu
eva
                if event.key == pygame.K_RIGHT:
                    change = "RIGHT" #hace que se mueva en dirección a la de
recha
                if event.key == pygame.K_LEFT:
                    change = "LEFT" #dirección izquierda
                if event.key == pygame.K_UP:
                    change = "UP" #dirección hacia arriba
                if event.key == pygame.K_DOWN:
                    change = "DOWN" #dirección hacia abajo
            if change == "RIGHT":
                snake_pos[0] += 10 #efecto de que se mueve
            if change == "LEFT":
                snake_pos[0] -= 10
            if change == "UP":
                snake_pos[1] -= 10 #resta porque nos acercamos a la y
            if change == "DOWN":
```

```
                snake_pos[1] += 10 #sumas porque nos alejamos de y

            snake_body.insert(0,list(snake_pos)) #añadimos al cuerpo lo que
hemos sumado o restado al cuerpo
            if snake_pos == food_pos: #la cabeza de la serpiente está en la
misma posición que la de la comida
                food_pos = food() #si esto sucede la comida se irá a otro si
tio
                score += 1
                print(score)
            else:
                snake_body.pop()

            play_surface.fill((0,0,0)) #pintamos la pantalla

            for pos in snake_body:#creamos el cuerpo de la serpiente
                pygame.draw.rect(play_surface,(200,200,200),pygame.Rect(pos[
0],pos[1],10,10))

            pygame.draw.rect(play_surface,(169,6,6),pygame.Rect(food_pos[0],
food_pos[1],10,10)) #dibujamos la manzana
            text = font.render(str(score),0,(200,60,80)) #marcamos la puntua
ción
            play_surface.blit(text,(480,20))

            if score < 10: #con una puntuación menor a 10 va lento
                fps.tick(10)
            if score >= 10: #con una puntuación mayor o igual la velocidad a
umenta
                fps.tick(20)

            if snake_pos[0] <= 0 or snake_pos[0] >= 500:
                run = False
                print("YOU LOSE")
            if snake_pos[1] <= 0 or snake_pos[1] >= 500:
                run = False
                print("YOU LOSE")

            pygame.display.flip()


main()

pygame.quit()
```

Código 1. Juego snake.

Octubre 2020	Enero 2021
– Computadora	– Computadora
– Investigación de los juegos arcade	– Investigación de los juegos arcade
– Instalación de Python	– Instalación de Python
– Aprender a programar Python	– Instalación de codificador. – (visual studio code)
– Conocer las bases del videojuego	– Aprender programación de Python
– Implementar una buena arquitectura de software	– Conocimiento de las bases de videojuego
	– Arquitectura de software
	– Conocer los colores en Python.
	– Instalar librería Pygame

Tabla 1. Tabla comparativa de recursos informáticos.

Octubre 2020	Enero 2021
0$	0$

Tabla 2. Tabla comparativa de costos asociados.

Nombre actividad	Fecha de inicio	Duración en días	Fecha fin
Diseño del juego	26-oct	2	28-oct
Clase jugador	28-oct	3	31-oct
Clase manzana	31-oct	1	01-nov
Clase juego	02-nov	1	03-nov
Clase aplicación	04-nov	3	07-nov
Clase main	09-nov	2	11-nov
Ejecución	11-nov	1	12-nov

Tabla 3. Tabla en relación al diagrama de Gantt, octubre-noviembre 2020.

Nombre actividad	Fecha de inicio	Duración en días	Fecha fin
Diseño del juego	28-dic	2	29-dic
Clase jugador	02-ene	2	03-ene
Clase manzana	03-ene	1	03-ene
Clase juego	04-ene	1	04-ene
Clase aplicación	05-ene	1	05-ene
Ejecución	07-ene	1	07-ene

Tabla 4. Tabla en relación al diagrama de Gantt, enero 2021.

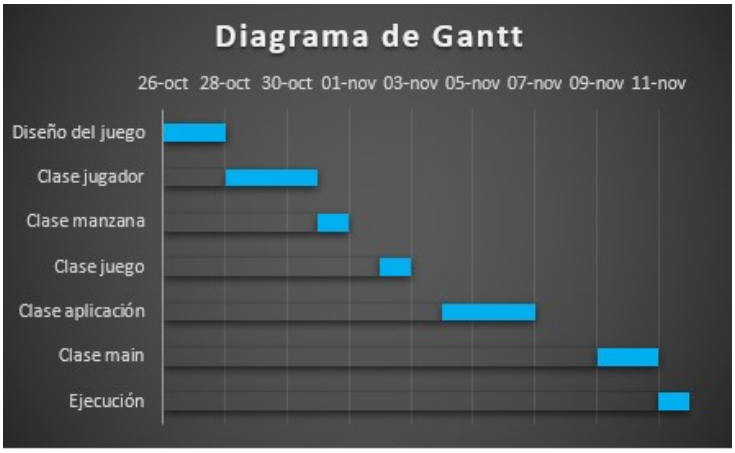

Diagrama de Gantt 1. Octubre-noviembre 2020.

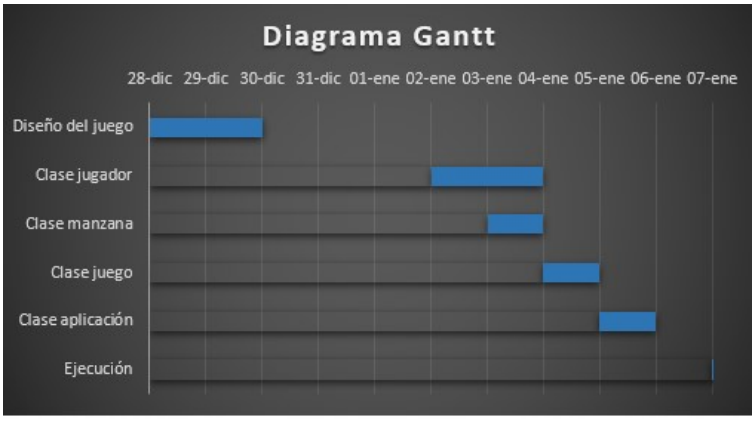

Diagrama de Gantt 2. enero 2021.

Canal de YouTube:

https://www.youtube.com/channel/UCJsj2--BFNmXDdiQhgfyhFw/featured

Repositorio GitHub:

[Rivera-Frida/Proyecto-Final (github.com)]

Guía rápida de usuario:

Para poder utilizar el juego, tenemos que tener Python instalado, así como sugerencia el editor de texto Visual Studio Code,

- Abrimos, desde el computador en donde se quiera jugar, símbolo del sistema, compilamos y corremos el programa.
- Se abrirá la ventana, el juego empieza.
- La comida aparecerá aleatoriamente por la pantalla, para llegar a ella e ir subiendo de puntuación, manejaremos a la serpiente con las flechas en el teclado, para movernos en la dirección querida.
- Se termina la partida cuando la serpiente sale de la ventana y aparecerá tu puntuación junto con el mensaje “you lose”.
- Para volver a jugar, vuelves a correr el programa desde el símbolo del sistema.