NOMBRE:

JUAN CARLOS PÉREZ

ASIGNATURA:

LÓGICA DE PROGRAMACIÓN

FECHA:

02/03/2025

TÍTULO:

EL IMPACTO DE LAS NUEVAS TECNOLOGÍAS EN LA SOCIEDAD: VISUALIZACIÓN DEL FUTURO

ÍNDICE

SELECCIÓN DEL PROGRAMA A DESARROLLAR / GENERACIÓN DE DIAGRAMAS FUNCIONALES Y ARQUITECTURA DE SOFTWARE 3
1. Tipos de Diagramas 3
Diagrama de caso de uso básico 3
Diagrama de Arquitectura 3
2. Escoger el Software a Desarrollar 3
Resolución de Problemas 3
Objetivo principal 3
Problemas específicos a resolver: 3
Objetivos educativos del proyecto: 4
3. Diagrama de caso de uso 4
INICIO DEL DESARROLLO DE SOFTWARE / CONFIGURACIÓN DEL ENTORNO 6
1. Configurar Github para cargar el Desarrollo 6
2. Diagrama de Flujo en función de las funcionalidades identificadas 6
3. Ambiente de Desarrollo 7
4. Avance de codificación del desarrollo del Software 7
DESARROLLO DEL PROGRAMA SELECCIONADO 7
1. Estructuras Lógicas 7
2. Estructuras Repetitivas 8
3. Comentarios de funcionalidades para mejor entendimiento 8
APLICANDO TÉCNICAS DE PROGRAMACIÓN FUNCIONAL 8
Foros de discusión 8
LINK DE PRESENTACIÓN: 8
CONCLUSIÓN 8
REFERENCIAS 9

SELECCIÓN DEL PROGRAMA A DESARROLLAR / GENERACIÓN DE DIAGRAMAS FUNCIONALES Y ARQUITECTURA DE SOFTWARE

1. Tipos de Diagramas

Diagrama de caso de uso básico

Representa las interacciones principales entre los actores y el sistema.

Diagrama de Arquitectura

Muestra la organización y estructura de un sistema de software, donde se incluyen componentes e interacciones.

2. Escoger el Software a Desarrollar

El software para desarrollar será el juego de la serpiente.

Resolución de Problemas

Objetivo principal

El juego de la serpiente busca resolver el diseño y la implementación a través de la interactividad donde una serpiente se mueva por la pantalla, a su vez come las frutas y conforme va comiendo va creciendo, donde no debe chocar con las paredes o consigo misma.

Problemas específicos a resolver:

Movimiento de la serpiente: Implementar la lógica para que la serpiente se mueva de manera continua en la dirección indicada por el jugador.

Detección de colisiones: Detectar cuando la serpiente choca con las paredes o consigo misma, lo que termina el juego.

Generación de comida: Colocar la comida en posiciones aleatorias en la pantalla y asegurarse de que no aparezca en la misma posición que la serpiente.

Crecimiento de la serpiente: Aumentar la longitud de la serpiente cada vez que come comida.

Puntuación: Llevar un registro de la puntuación del jugador basada en la cantidad de comida que la serpiente ha comido.

Objetivos educativos del proyecto:

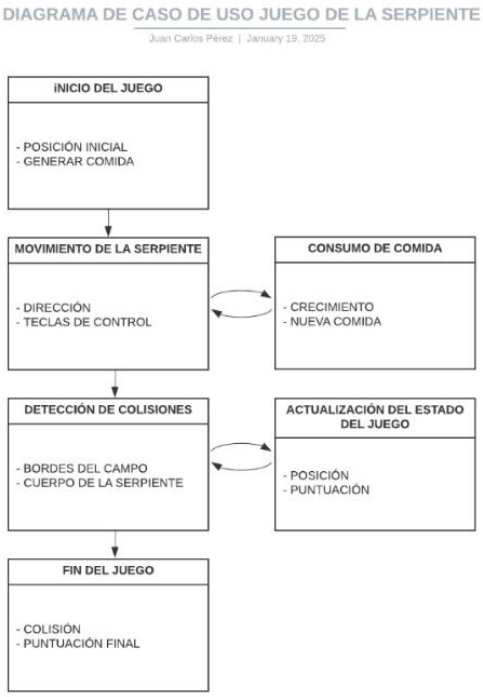
Lógica de programación: Aprender a implementar estructuras de control como bucles y condicionales.

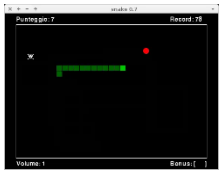
Manipulación de gráficos: Trabajar con gráficos básicos para dibujar la serpiente, la comida y el entorno del juego.

Gestión de eventos: Manejar la entrada del usuario para controlar la dirección de la serpiente.

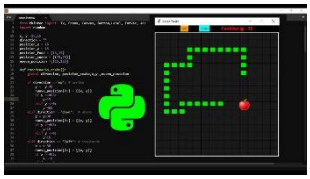
Resolución de problemas: Desarrollar habilidades para identificar y solucionar problemas lógicos y de diseño en el código.

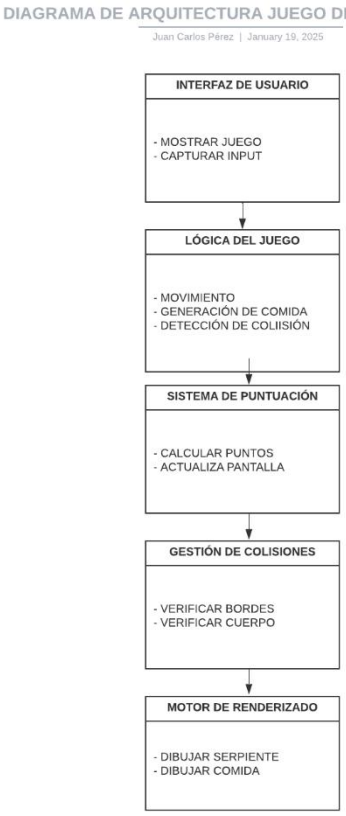
3. Diagrama de caso de uso





4. Diagrama de Arquitectura

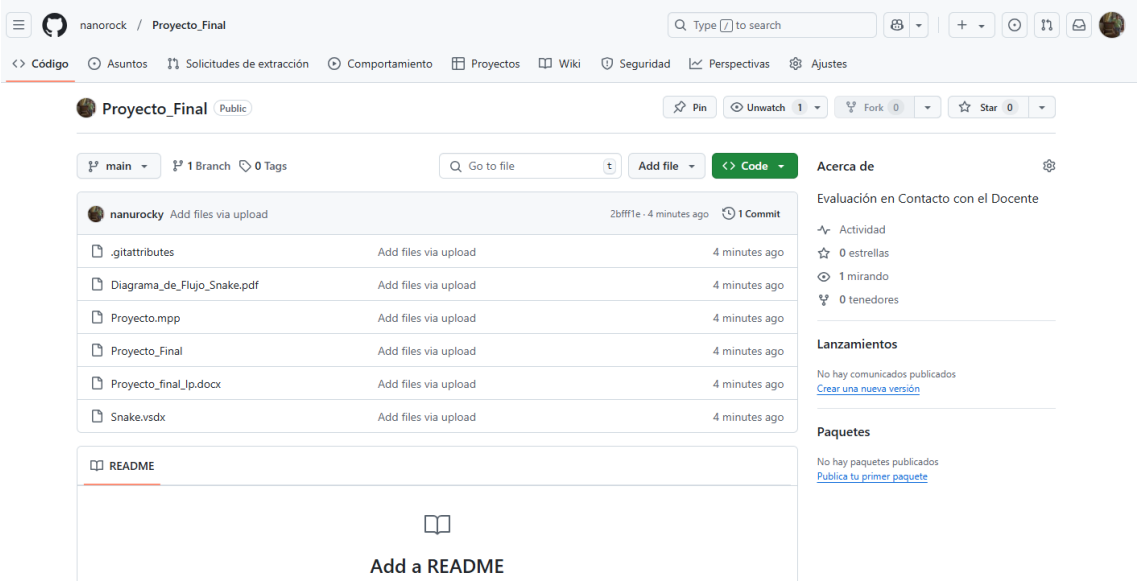




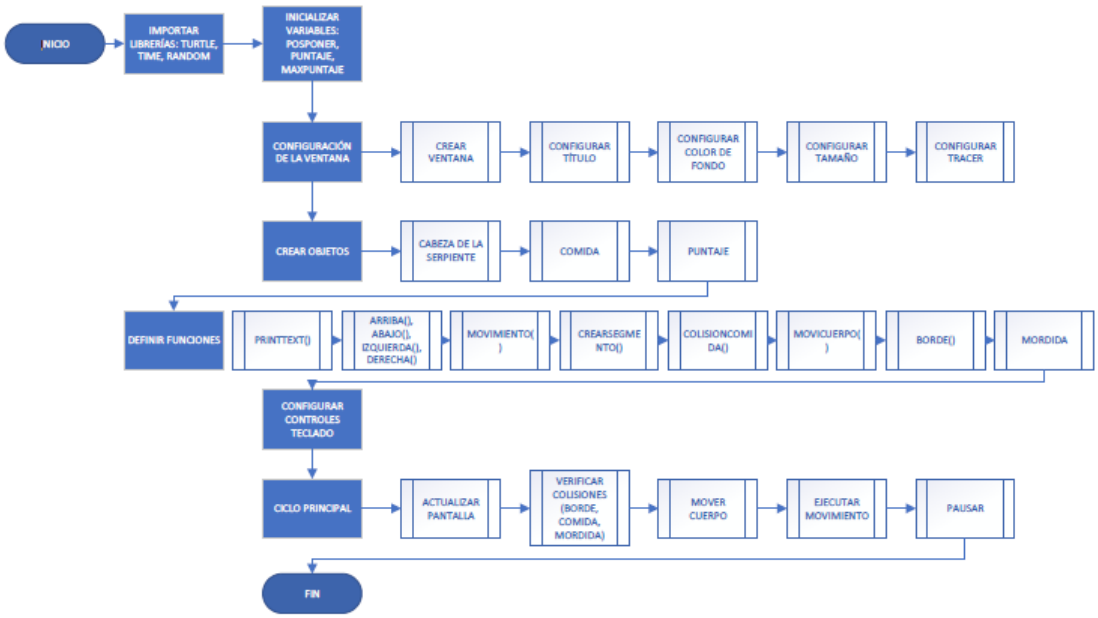
INICIO DEL DESARROLLO DE SOFTWARE / CONFIGURACIÓN DEL ENTORNO

1. Configurar Github para cargar el Desarrollo

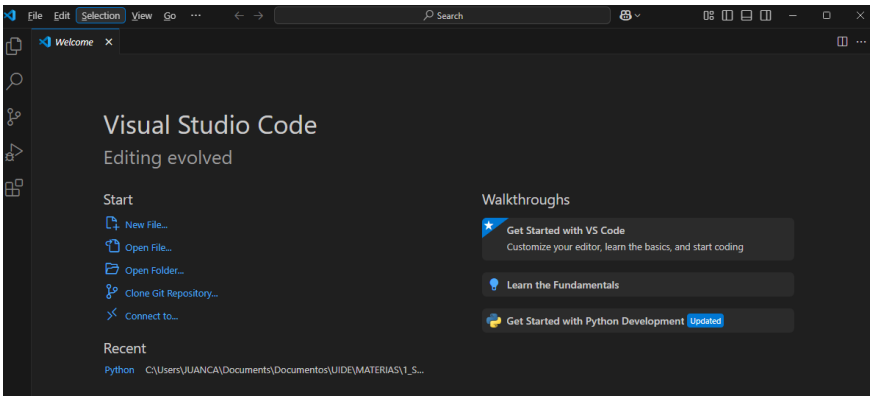
Link de GitHub: https://github.com/nanurocky/Proyecto_Integrador_Final.git



2. Diagrama de Flujo en función de las funcionalidades identificadas



3. Ambiente de Desarrollo



4. Avance de codificación del desarrollo del Software

C: > Users > JUANCA > Documents > Documentos > UIDE > MATERIAS > 1_SEMESTRE > LÓGICA_DE_PROGRAMACIÓN > Tareas > Proyecto_Integrador > Proyecto_Final

```
1  import turtle                    #La libreria turtle ayuda a construir la interfaz gráfica
2  import time                      #Para obtenier el tiempo
3  import random                    #Para usar numeros random
4
5  posponer = 0.1
6  puntaje = 0
7  maxPuntaje = 0
8
9  #Configuración
10 window = turtle.Screen()         #Crea una ventana nueva
11 window.title('Snake')            #Ponemos titulo
12 window.bgcolor('#353535')        #Color de fondo
13 window.setup(width=600,height=600)  #Redimensionar pantalla
14 window.tracer(0)                 #Ayuda a hacer la animación mas placentera
15
```

DESARROLLO DEL PROGRAMA SELECCIONADO

1. Estructuras Lógicas

```
#Ejecuta el movimiento
def movimiento():
    if cabeza.direction == 'up':
        y =  cabeza.ycor()
        cabeza.sety(y + 20)

    elif cabeza.direction == 'down':
        y =  cabeza.ycor()
        cabeza.sety(y - 20)

    elif cabeza.direction == 'left':
        x =  cabeza.xcor()
        cabeza.setx(x - 20)

    elif cabeza.direction == 'right':
        x =  cabeza.xcor()
        cabeza.setx(x + 20)
```

2. Estructuras Repetitivas

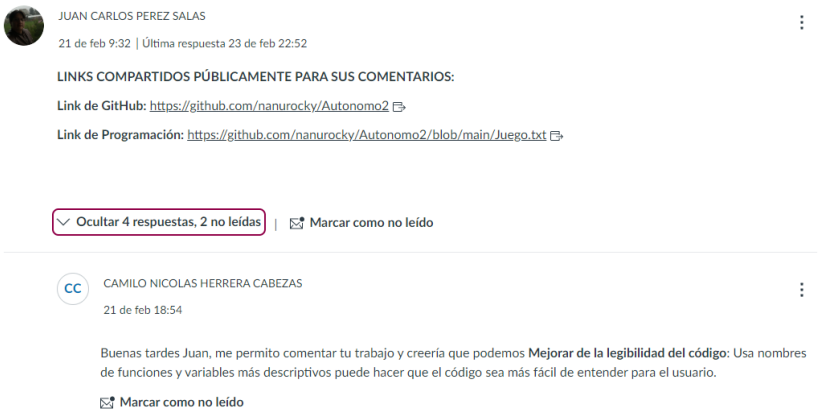
```
#Cada elemento sigue al anterior
#Exepto el primero
for segmento in range(totalSeg-1,0,-1):
    x = cuerpo[segmento-1].xcor()
    y = cuerpo[segmento-1].ycor()
    cuerpo[segmento].goto(x,y)
```

3. Comentarios de funcionalidades para mejor entendimiento

```
#Cabeza de la serpiente
cabeza = turtle.Turtle()              #Crea un objeto para mostrar en pantalla
cabeza.speed(0)                       #Se muestra al inciar
cabeza.shape('square')                #Se le asigna forma de cuadrado, por defecto es 20x20 pixeles
cabeza.color('#75C46D')               #Color a la cabeza
cabeza.penup()                        #Elimina el rastro del objeto
cabeza.goto(0,0)                      #Centra el objeto
cabeza.direction = 'stop'             #Asigna direccion, en este caso estatico
```

APLICANDO TÉCNICAS DE PROGRAMACIÓN FUNCIONAL

Foros de discusión



LINK DE PRESENTACIÓN:

https://www.canva.com/design/DAGgQCRu61Q/rSqj0l4syJcgAqF4oZH0sA/view?utm_content=DAGgQCRu61Q&utm_campaign=designshare&utm_medium=link2&utm_source=uniquelinks&utlId=h6bfc2374a3

CONCLUSIÓN

Con la implementación de este juego se pudo aprender la lógica de programación donde se desarrollaron habilitades en manipulación de efectos de movimientos debido a estructuras lógicas y de repetición donde se fomenta la resolución de problemas y el pensamiento crítico y nos preparamos para un desafío hacia el futuro mucho más complejo en cuanto al desarrollo de software se refiere.

REFERENCIAS

a. (Github, s.f.)
b. (Lucidchart, s.f.)
c. (Candia-Véjar, 2011)