

LET'S PLAY!

# HISTORIA

El juego de la culebrita, también conocido como Snake, se originó en un videojuego arcade competitivo de 1976 llamado Blockade desarrollado por Gremlin Inc. El objetivo era sobrevivir más tiempo que el otro jugador.

En 1978, Worm fue la versión para ordenador del posterior Snake.

En 1982, Chuck Sommerville publicó un juego llamado Snake Byte para Apple II.

En 1998, Taneli Armando creó Snake para el teléfono móvil Nokia 6110.

Snake se convirtió en un juego estándar pregrabado en los teléfonos Nokia.

El juego Snake se popularizó masivamente y se convirtió en un clásico.

# CARACTERÍSTICAS

- El jugador debe evitar que la serpiente choque con otros obstáculos y consigo misma.
- La dificultad aumenta a medida que la serpiente se alarga.
- El juego se basa en una línea con cabeza y cola que se hace más larga con cada trozo de comida ingerido.

# CODIFICACION

Dentro del código de SNAKE se tuvieron que importar bibliotecas como pygame que permite el diseño de juegos en 2D y Json para el intercambio y el almacenamiento de datos que esto nos va ayudar a almacenar el mejor puntaje.

Además se comenzó la configuración del juego designando el tamaño de la pantalla y las celdas, así como los colores.

```
Snake.py > ...
1 import pygame
2 import sys
3 import random
4 import json
5
6 # Configuración del juego
7 WIDTH, HEIGHT = 600, 400
8 CELL_SIZE = 20
9 WHITE = (255, 255, 255)
10 GREEN = (0, 255, 0)
11 RED = (255, 0, 0)
12 BLUE = (0, 0, 255)
13
```

# ALMACENAMIENTO DE PUNTAJES

Para el almacenamiento e intercambio de datos del registro y la carga del puntaje mas alto se crearon dos funciones (def y try para iniciar un bloque de código), una para cargar el puntaje mas alto y el otro para guardarlo, así para tener un record que romper y ver quien tiene el mejor puntaje. Como se muestra la función se creo con condicionales(if), manejo de excepciones para para que siga funcionando (except), (with) para abrir y cerrar archivos y return para devolver valores de la función.

```
14 # Cargar el puntaje más alto
15 def load_high_score():
16     try:
17         with open("high_score.json", "r") as file:
18             return json.load(file)
19     except (FileNotFoundException, json.JSONDecodeError):
20         return {"name": "", "score": 0}
21
22 # Guardar el puntaje más alto
23 def save_high_score(name, score):
24     high_score = load_high_score()
25     if score > high_score["score"]:
26         with open("high_score.json", "w") as file:
27             json.dump({"name": name, "score": score}, file)
28
```

# FUNCION PRINCIPAL DEL JUEGO

Aquí se creó un bucle que permite controlar la función principal del juego, es decir, que permite que partes de código se ejecuten y en qué orden, donde dentro del bucle, se programó la comida, la culebra, las direcciones que puede tomar la culebra, que hace en caso de comer la comida que es crecer, que hace si choca con los bordes o consigo misma que es perder el juego, se programó para que se dibuje la comida en lugares aleatorios, se programó para mostrar la suma de puntaje durante el juego. Para lograr esto se aplicaron tuplas, bucles, condicionales y listas.

```
29 # Función principal del juego
30 def game_loop(player_name):
31     pygame.init()
32     screen = pygame.display.set_mode((WIDTH, HEIGHT))
33     clock = pygame.time.Clock()
34     font = pygame.font.Font(None, 30)
35
36     snake = [(100, 100)]
37     direction = (CELL_SIZE, 0)
38     food = (random.randint(0, WIDTH // CELL_SIZE - 1) * CELL_SIZE,
39             random.randint(0, HEIGHT // CELL_SIZE - 1) * CELL_SIZE)
39     score = 0
40
41     while True:
42         screen.fill(WHITE)
43
44         for event in pygame.event.get():
45             if event.type == pygame.QUIT:
46                 save_high_score(player_name, score)
47                 pygame.quit()
48                 sys.exit()
49             elif event.type == pygame.KEYDOWN:
50                 if event.key == pygame.K_UP and direction != (0, CELL_SIZE):
51                     direction = (0, -CELL_SIZE)
52                 elif event.key == pygame.K_DOWN and direction != (0, -CELL_SIZE):
53                     direction = (0, CELL_SIZE)
54                 elif event.key == pygame.K_LEFT and direction != (CELL_SIZE, 0):
55                     direction = (-CELL_SIZE, 0)
56                 elif event.key == pygame.K_RIGHT and direction != (-CELL_SIZE, 0):
57                     direction = (CELL_SIZE, 0)
58
59         # Mover la culebra
60         new_head = (snake[0][0] + direction[0], snake[0][1] + direction[1])
61
62         if new_head in snake or not (0 <= new_head[0] < WIDTH and 0 <= new_head[1] < HEIGHT):
63             save_high_score(player_name, score)
64             break
65
66         snake.insert(0, new_head)
67
68         if new_head == food:
69             score += 10
70             food = (random.randint(0, WIDTH // CELL_SIZE - 1) * CELL_SIZE,
71                     random.randint(0, HEIGHT // CELL_SIZE - 1) * CELL_SIZE)
72         else:
73             snake.pop()
74
75         # Dibujar la culebra y la comida
76         for segment in snake:
77             pygame.draw.rect(screen, GREEN, (*segment, CELL_SIZE, CELL_SIZE))
78         pygame.draw.rect(screen, RED, (*food, CELL_SIZE, CELL_SIZE))
79
80         # Mostrar el puntaje
81         score_text = font.render(f"Puntaje: {score}", True, BLUE)
82         screen.blit(score_text, (10, 10))
83
84
85         pygame.display.flip()
86         clock.tick(10)
```

# MENU

```
88 # Menú principal
89 def main_menu():
90     pygame.init()
91     screen = pygame.display.set_mode((WIDTH, HEIGHT))
92     font = pygame.font.Font(None, 40)
93     clock = pygame.time.Clock()
94     input_box = pygame.Rect(200, 150, 200, 40)
95     color_inactive = pygame.Color('lightskyblue3')
96     color_active = pygame.Color('dodgerblue2')
97     color = color_inactive
98     active = False
99     text = ''
100
101    while True:
102        screen.fill(WHITE)
103
104        title = font.render("SNAKE GAME", True, GREEN)
105        screen.blit(title, (WIDTH / 2 - 80, 50))
106
107        prompt = font.render("Ingresa tu nombre:", True, BLUE)
108        screen.blit(prompt, (200, 100))
109
110        pygame.draw.rect(screen, color, input_box, 2)
111        text_surface = font.render(text, True, BLUE)
112        screen.blit(text_surface, (input_box.x + 5, input_box.y + 5))
113
114        high_score = load_high_score()
115        high_score_text = font.render(f"Mejor Puntaje: {high_score['name']} - {high_score['score']}", True, RED)
116        screen.blit(high_score_text, (150, 250))
117
118
119        for event in pygame.event.get():
120            if event.type == pygame.QUIT:
121                pygame.quit()
122                sys.exit()
123            elif event.type == pygame.MOUSEBUTTONDOWN:
124                if input_box.collidepoint(event.pos):
125                    active = not active
126                else:
127                    active = False
128            color = color_active if active else color_inactive
129            elif event.type == pygame.KEYDOWN:
130                if active:
131                    if event.key == pygame.K_RETURN:
132                        game_loop(text)
133                    elif event.key == pygame.K_BACKSPACE:
134                        text = text[:-1]
135                    else:
136                        text += event.unicode
137
138            pygame.display.flip()
139            clock.tick(30)
140
141    if __name__ == "__main__":
142        main_menu()
```

Para el menu se aplico otro bucle, donde se programo la pantalla de inicio, el registro del nombre de usuario y la demostracion del record o puntaje mas alto; esto igualmente se logro a traves de bucles, funciones, listas y condicionales donde al final se deja el main menu para la ejecucion del menu del juego.

