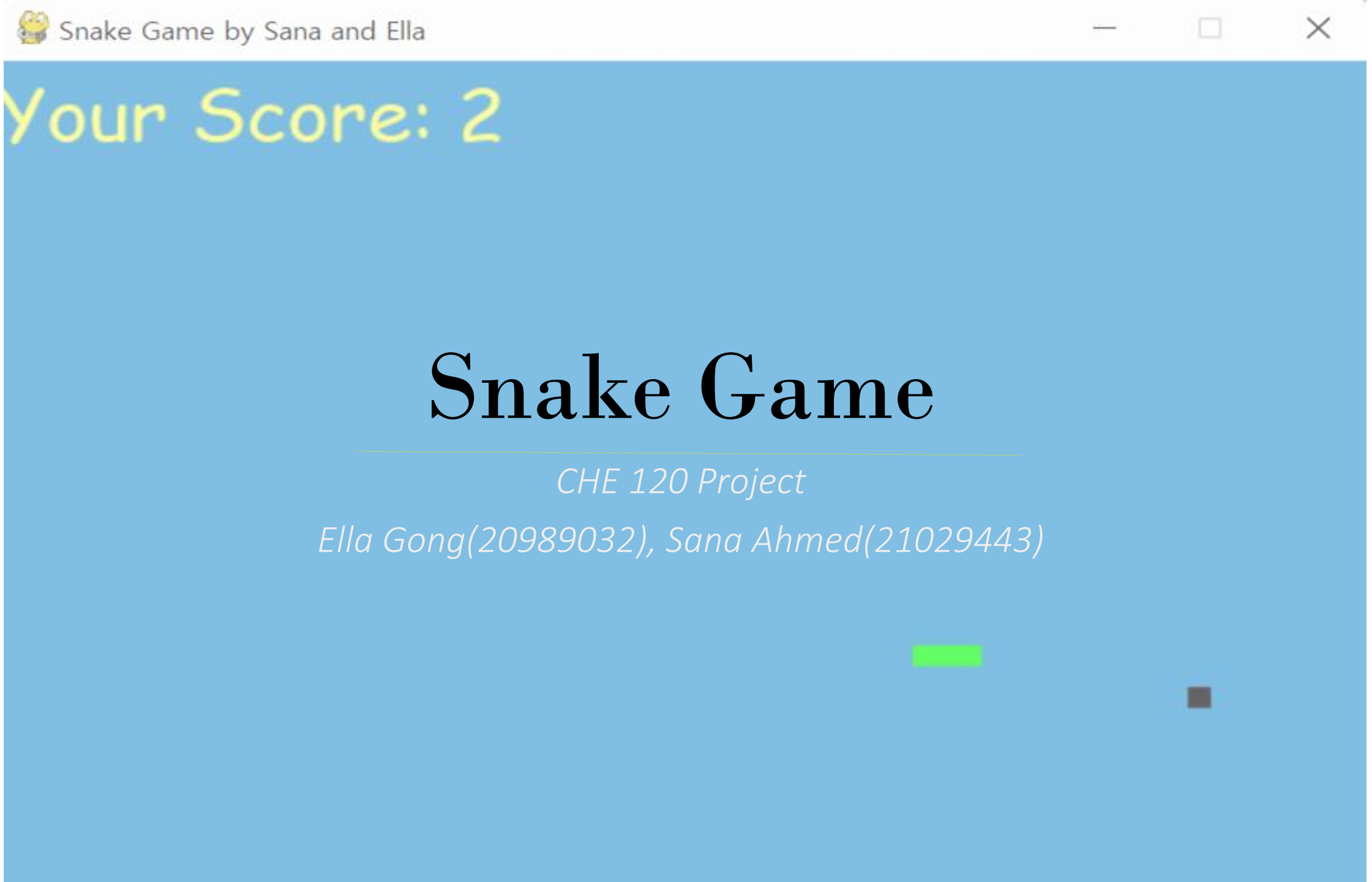
Snake Game by Sana and Ella
Your Score: 2
Snake Game
CHE 120 Project
Ella Gong(20989032), Sana Ahmed(21029443)

Table of content

Snake Game

- A game that controls a snake that gets longer as it eats the food.
- The player loses if the snake bumps into itself or a wall.
- The player wins if the snake fills out the whole screen with its body.

- The very first Snake-type game was an arcade game called Blockade, created by Gremlin way back in 1976.
- Snake first appeared on a Nokia device in 1997 on the Nokia 6110.

https://blogs.windows.com/devices/2012/02/02/snake-charmed-10-fascinating-facts-about-the-worlds-most-popular-game/

Original Code with comments

Source by

https://www.edureka.co/blog/snake-game-with-pygame/

```
WEG, Ella Gong
WSA, Sana Ahmed

#import required modules WSA
import pygame Wused to create game and graphics WSA
import time #used to time track the time WSA
import random #used to randomize food location WSA

pygame.init() # initialize WEG

#set colour ranges in pygame WSA
white = (255, 255, 255)
yellow = (255, 255, 102)
black = (0, 0, 0)
red = (213, 50, 80)
green = (0, 255, 0)
blue = (50, 153, 213)

#set display width and height WSA
dis_width = 600
dis_height = 400

dis = pygame.display.set_mode((dis_width, dis_height)) # create a screen WEG
pygame.display.set_caption('Snake Game by Edureka') # show display caption WSA

#initialize variable to track amount of time taken WSA
clock = pygame.time.Clock()

snake_block = 10 # snake's size WEG
snake_speed = 15 # snake's speed WEG

font_style = pygame.font.SysFont("bahnschrift", 25) # font is in 25 size with font "bahnschrift" WEG
score_font = pygame.font.SysFont("comicsansms", 35) # font is in 35 size with font "comicsansms" WEG
```

```
#define score function to display score WSA
def Your_score(score):
    value = score_font.render("Your Score: " + str(score), True, yellow) #assign variable value, to display core message in yellow
    dis.blit(value, [0, 0]) #block bit transfer the score at index [0,0] of display WSA

#define snake function to create snake WSA
def our_snake(snake_block, snake_list):
    for x in snake_list: #for each element in the snake body WSA
        pygame.draw.rect(dis, black, [x[0], x[1], snake_block, snake_block]) # draw a rectangle -> our snake WEG

#define message function to create colour and style of font WSA
def message(msg, color):
    mesg = font_style.render(msg, True, color) # message is in color with font_style WEG
    dis.blit(mesg, [dis_width / 6, dis_height / 3]) # message is printed on that location WEG

#define gameLoop function to restart game when called WSA
def gameLoop():
    game_over = False  # If True, game is over WEG
    game_close = False # if True, game over screen appears WEG

    x1 = dis_width / 2 # initial location in x-axis WEG
    y1 = dis_height / 2 # initial location in y-axis WEG

    x1_change = 0 # change made in moving in x-axis WEG
    y1_change = 0 # change made in moving in y-axis WEG

    #intialize body of snake as a list WSA
    snake_List = []
    Length_of_snake = 1 #length of snake begins with 1 WEG

    foodx = round(random.randrange(0, dis_width - snake_block) / 10.0) * 10.0 # food's location is randomly chosen except for current location WEG
    foody = round(random.randrange(0, dis_height - snake_block) / 10.0) * 10.0 # food's location is randomly chosen except for current location WEG
```

```
while game_close == True: # if the game over screen appears WEG
    dis.fill(blue) # the background color is blue WEG
    message("You Lost! Press C-Play Again or Q-Quit", red) #prints lose message on scren WSA
    Your_score(Length_of_snake - 1) #calls function of score, -1 to negate snake head WSA
    pygame.display.update() # update any changes made to screen WEG

    for event in pygame.event.get(): #iterating through each event in pygame WSA
        if event.type == pygame.KEYDOWN: # if you press a key: WEG
            if event.key == pygame.K_q: # if you press keyboard "q", game is closed WEG
                game_over = True # the game is over WSA
                game_close = False #the game is over but screen still runs WSA
            if event.key == pygame.K_c: # if press keyboard "c", game loops WEG
                gameLoop()

for event in pygame.event.get(): #iterating through each event in pygame WSA
    if event.type == pygame.QUIT:#if event is QUIT WSA
        game_over = True #the game is over WSA
    if event.type == pygame.KEYDOWN: # event is placed when you press a keyboard WEG
        if event.key == pygame.K_LEFT: # if push left, move to -x axis WEG
            x1_change = -snake_block
            y1_change = 0 # y-axis does not change when moving left WSA
        elif event.key == pygame.K_RIGHT: # if push right, move to +x axis WEG
            x1_change = snake_block
            y1_change = 0 # y-axis does not change when moving right WSA
        elif event.key == pygame.K_UP: # if push up, move to +y axis WEG
            y1_change = -snake_block
            x1_change = 0 # x-axis does not change when moving up WSA
        elif event.key == pygame.K_DOWN: # if push down, move to -y axis WEG
```

```
    if x1 >= dis_width or x1 < 0 or y1 >= dis_height or y1 < 0: # if snake hits the boundary, game over screen appears WEG
        game_close = True #the game ends WSA
    x1 += x1_change # x1 is updated by adding the x1_change in x-axis WEG
    y1 += y1_change # y1 is updated by adding the x1_change in x-axis # x1, y1 is current coordinate of snake WEG
    dis.fill(blue) # background is blue color WEG

    pygame.draw.rect(dis, green, [foodx, foody, snake_block, snake_block]) #pygame draw green rectangle for food and snake WSA
    snake_Head = [] # snake head starts with empty list WEG
    snake_Head.append(x1) # add x1 to snake_head list WEG
    snake_Head.append(y1) # add y1 to snake_head list WEG
    snake_List.append(snake_Head) # add snake_head to snake_List list WEG
    if len(snake_List) > Length_of_snake: #if the length of the snake body in list is greater than 1 WSA
        del snake_List[0] # delete the head of the snake, which is index 0 WSA

    for x in snake_List[:-1]: #for each element in snake body WSA
        if x == snake_Head: #if any element of the body hits the snakes head WSA
            game_close = True # the game ends WSA

    our_snake(snake_block, snake_List) # calls snake_function, the snake body WSA
    Your_score(Length_of_snake - 1) # calls score function, -1 to negate the snake head WSA

    pygame.display.update() # update any changes made to screen WEG

    if x1 == foodx and y1 == foody: # if snake's location matches with food's location, WEG
        foodx = round(random.randrange(0, dis_width - snake_block) / 10.0) * 10.0 # food's location is randomly chosen except for current location WEG
        foody = round(random.randrange(0, dis_height - snake_block) / 10.0) * 10.0 # food's location is randomly chosen except for current location WEG
        Length_of_snake += 1 # length of snake increases by factor of 1 WEG

    clock.tick(snake_speed) #calls the function of the snake speed WSA

pygame.quit() # uninitialize pygame WEG
```

Variation

New Rule

New Rules Added

1. Music is on

-Sound effects are added including when eating an apple, bumping into obstacles etc

2. Apple has a random sprinkling colour.

3. You spawn randomly when a snake eats an apple.

Final Code(change)

3 different variations

Bgm.mp3 Crunch.wav Boing.wav

1. Music

```
def gameLoop():
    music = pygame.mixer.music.load('bgm.mp3') # music file "bgm.mp3" is loaded and defined to music WEG
    pygame.mixer.music.play(-1) # music is set as background of the game WEG
    crunch_sound = pygame.mixer.Sound('crunch.wav') # "crunch.wav" sound effect is defined as crunch_sound WEG
    bump_sound = pygame.mixer.Sound('boing.wav') # "boing.wav" sound effect is defined as bump_sound WEG
```

```
if x1 >= dis_width or x1 < 0 or y1 >= dis_height or y1 < 0:
    bump_sound.play() # when the snake bumps into wall, bump_sound is played WEG
    game_close = True
```

```
if x1 == foodx and y1 == foody:
    foodx = round(random.randrange(0, dis_width - snake_block) / 10.0) * 10.0
    foody = round(random.randrange(0, dis_height - snake_block) / 10.0) * 10.0
    crunch_sound.play() # if the snake eats the food, crunch_sound is played WEG
    Length_of_snake += 1
```

2. Color 

```
def food_color():
    color_list = [white, yellow, black, red, green, blue]
    color = random.choice(color_list)
    return color  # return an randomly flashing color
```

```
pygame.draw.rect(dis, food_color(), [foodx, foody, snake_block, snake_block])
snake_Head = []  # food color is decided by food_color() function, randomly flashing colors
snake_Head.append(x1)
snake_Head.append(y1)
snake_List.append(snake_Head)
if len(snake_List) > Length_of_snake:
    del snake_List[0]
```

3. Randomly spawned when scoring

```
if x1 == foodx and y1 == foody:
    foodx = round(random.randrange(0, dis_width - snake_block) / 10.0) * 10.0
    foody = round(random.randrange(0, dis_height - snake_block) / 10.0) * 10.0
    crunch_sound.play() # if the snake eats the food, crunch_sound is played WEG
    Length_of_snake += 1

    x1 = round(random.randrange(0,dis_width - foodx) /10.0) * 10.0 # snake will be located in random space except for
    y1 = round(random.randrange(0,dis_height - foody) /10.0) * 10.0 # snake will be located in random space except for
                                                                    food location WEG
                                                                     food location WEG
```

Github Repository

- User ID : Ellagong0822
- Repository name : CHE120-Project