Multimedia Systems

A 2D Snake Game

Under Dr. Priyambada Subudhi

Group No: 14

Group Members:

Anirudh Jakhotia – S20190010007
Harish Mullagura – S20190010124
Neeraj Dusa – S20190010047
Rakesh Ganeshula – S20190010052

Abstract:

The problem is to design a Snake Game which provides the following functionalities:

- Snake can move in a given direction and when it eats the food, the length of snake increases.
- When the snake crosses itself, the game will be over.

Slytherin is a 2D snake game made using the pygame module. Some of its main features are :

- Main menu
- Pause functionality
- Highest score counter
- Runtime score
- Game music

Problem Statement:

To write a program in python using a pygame module that will animate a 2D Snake Game using many multimedia functionalities like images, audio and video.

Problem Solution:

By creating a 2D snake game using pygame module, it is beginner-friendly, making it a great platform for all adults and kids to like and the code written in pygame is kept simple which can run across many platforms and operating systems.

Requirements:

- Python - Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

- Pygame - Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language

- Vscode Editor - A streamlined code editor with support for development operations like debugging, task running, and version control.

Libraries:

- import pygame
- Import random
- from pygame.locals import *
- from pygame import mixer

Details about libraries:

- Pygame.locals - This module contains various constants used by Pygame. However, an application can use pygame. locals to include only the Pygame constants with a from pygame.

- from pygame import mixer- In order to play music/audio files in pygame , pygame.mixer is used. This module contains classes for loading Sound objects and controlling playback

- import pygame- Pygame is an open-source Python library for making multimedia applications like games built on top of the excellent SDL library.

- Import random - This module can be used to perform random actions such as generating random numbers, print random values for a list or string, etc.

- pygame. init()- Initialize all imported pygame modules.

How to execute the Files:

- Extract the zip folder of the code base.
- Install python on your computer. Link
- Install pygame using the command "pip install pygame".
- Move to the code directory using "cd Slytherin"
- Run the python file using the command "python slytherin.py"

Details about Files:

- Slytherin.py: Main Driver Program

- Images:
  - Apple.png
  - Icon.png
  - Pause.png
  - Start.png

- Sounds:
  - GameMusic.mp3
  - GameOver.mp3

Functions used for PyGame:

Code Snippets:

- def snake(block_width,block_height,SnakeList):
  It initializes and creates a single rectangle block snake.

pygame.draw.rect(): This function is used to draw a rectangle. It takes the surface, color, and pygame Rect object as an input parameter and draws a rectangle on the surface mainly for creating the snake.

Pygame.display() - This function sets the display mode in pygame and creates a visible image surface on the monitor. This surface can either cover the full screen, or be windowed on platforms that support a window manager.

- **Initialization :** This step firstly initializes all the pygame modules and then specifies the window size that needs to be created. It specifies the path of each image in the game with suitable audio path specific to each image.

```
pygame.init()
Win_Size = [800,500]
iconPath = 'images/icon.png'
icon = pygame.image.load(iconPath)
Display = pygame.display.set_mode(Win_Size)
pygame.display.set_caption("Snooby")
pygame.display.set_icon(icon)
clock = pygame.time.Clock()
```

```
mixer.pre_init(44100, -16, 2, 512)

gameMusicPath = 'sounds/GameMusic.mp3'
gameOverPath = 'sounds/GameOver.mp3'

mixer.music.load(gameMusicPath)
gameover_sound = mixer.Sound(gameOverPath)
```

```
startScreenPath = 'images/start.png'
ApplePath = 'images/Apple.png'
pausedPath = 'images/pause.png'

startScreen = pygame.image.load(startScreenPath)
Apple = pygame.image.load(ApplePath)
paused = pygame.image.load(pausedPath)
```

- **Start_Screen()** - It creates a surface object which displays the start screen display object on the initial window surface object. This is mainly the game call.

```
def Start_Screen():
    StartLoop = True

    while StartLoop == True:
        Display.fill(white)
        Display.blit(startScreen, [0,0])


        pygame.display.update()

        for event in pygame.event.get():
            if event.type == QUIT:
                pygame.quit()
                quit()

            if event.type == KEYDOWN:
                if event.key == K_c:
                    StartLoop = False

                if event.key == K_q:
                    pygame.quit()
                    quit()
```

- **Pause_Screen() –** When we call the pause function, we fade the music after 3000ms (3 sec) and show the pause frame. User has 2 options: continue (press c) and quit (press q). All the events are stored in an event queue and then they are initiated according to the calling order.

```
def Pause_Screen():
    Pause = True
    mixer.music.fadeout(1000)
    while Pause:

        for event in pygame.event.get():
            if event.type == KEYDOWN:
                if event.key == K_c:
                    mixer.music.play(-1)
                    Pause = False

                if event.key == K_q:
                    pygame.quit()
                    quit()


        Display.blit(paused,[0,0])
        pygame.display.update()
        clock.tick(5)
```

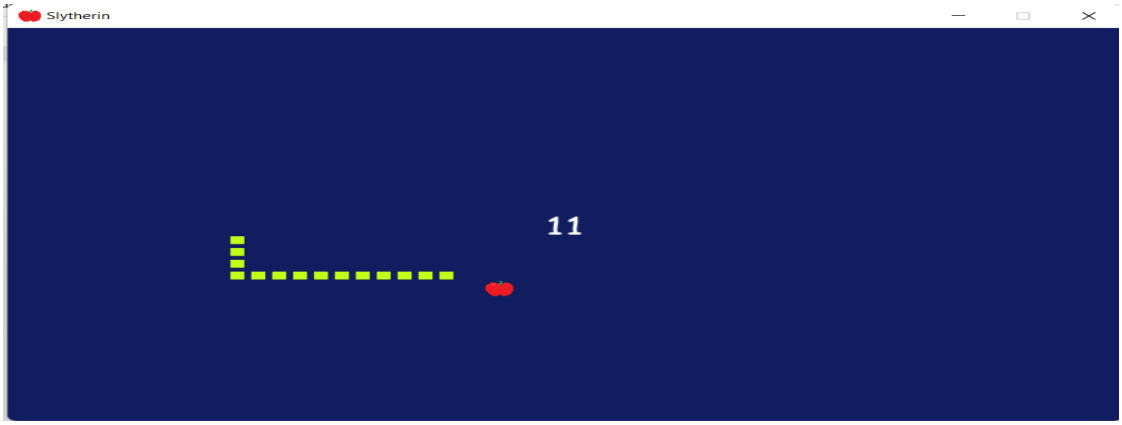
➢ **Game_loop() –** It is used for looping the game to start the next round which includes all game over and while playing game conditions like wall collisions, snake collision with apple, self-collision, etc.

```
def Game_Loop():

    ##Snake_Object
    SnakeList = []
    SnakeLength = 3


    pos_x = Win_Size[0]/2
    pos_y = Win_Size[1]/2
    pos_x_change = 0
    pos_y_change = 0
    snake_width = 10
    snake_height = 10
    snake_step = 15

    ##Apple_Object

    apple_width = 20
    apple_height = 20
    randApple_x = round(random.randrange(0,Win_Size[0]-apple_width))
    randApple_y = round(random.randrange(0,Win_Size[1]-apple_height))
```

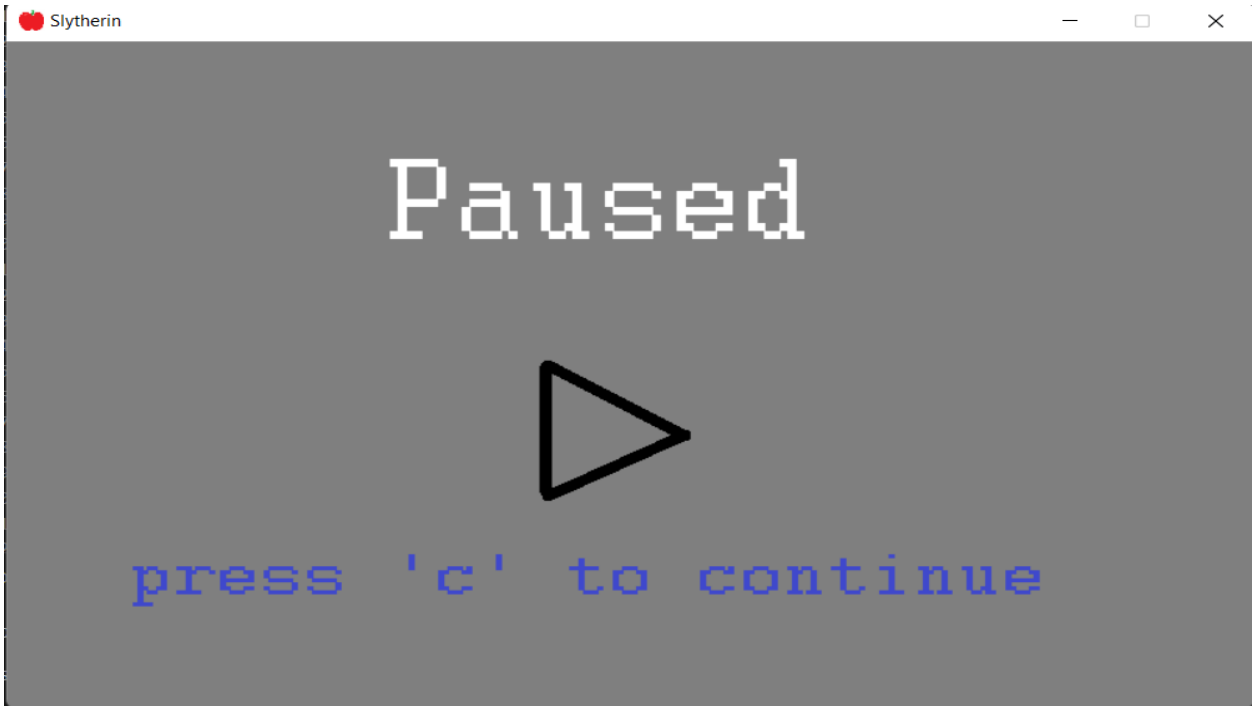
Gameplay Screenshots:

1) Start Screen

2) Gameplay



3) Paused Screen



4) Game Over

———————————— THANK YOU ————————————