# Multimedia Systems

# A 2D Snake Game

➢ Under Dr Priyambada Subudhi

## 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 snake crosses itself, the game will over. Slytherin is a 2D snake game made using 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 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

- 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 a value 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 in 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:

- def snake(block_width,block_height,SnakeList):

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.

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.

- ➢ Start_Screen()

## Game call...

- ➢ Pause_Screen()

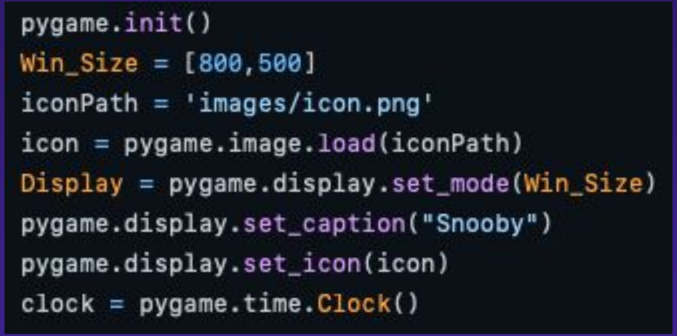
## Pauses the game using the key “p”.

- ➢ Game_loop()

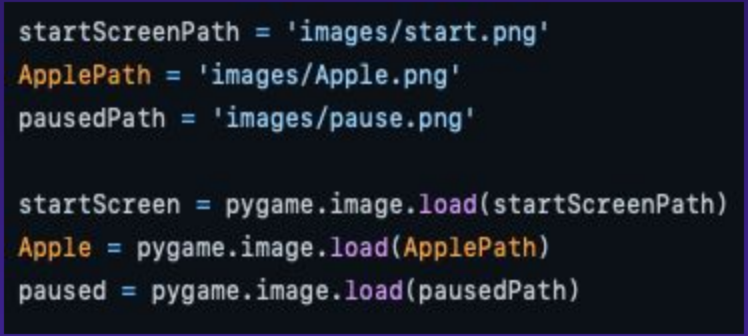
## It is the main loop for starting the game.

# Code Snippets

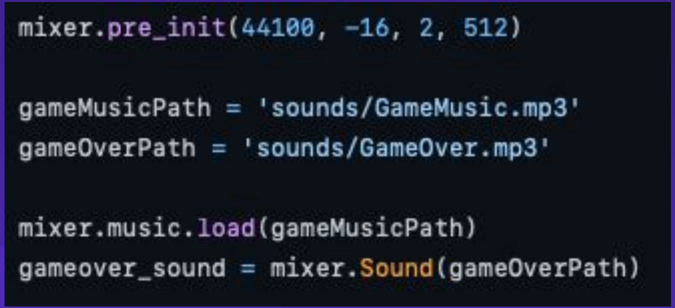
# Initialisation



```python
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()
```

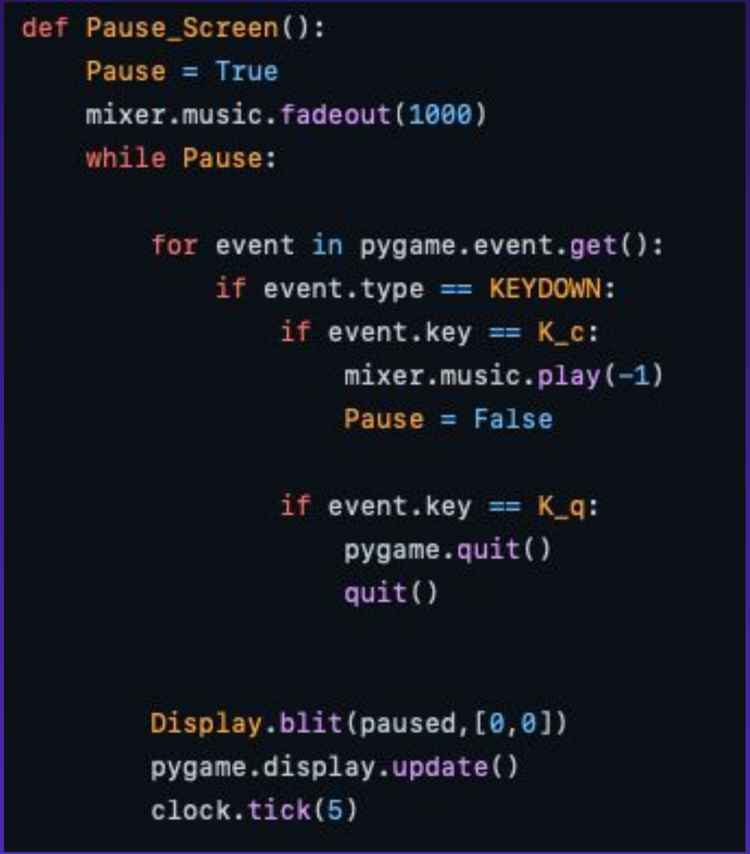


```python
startScreenPath = 'images/start.png'
ApplePath = 'images/Apple.png'
pausedPath = 'images/pause.png'

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



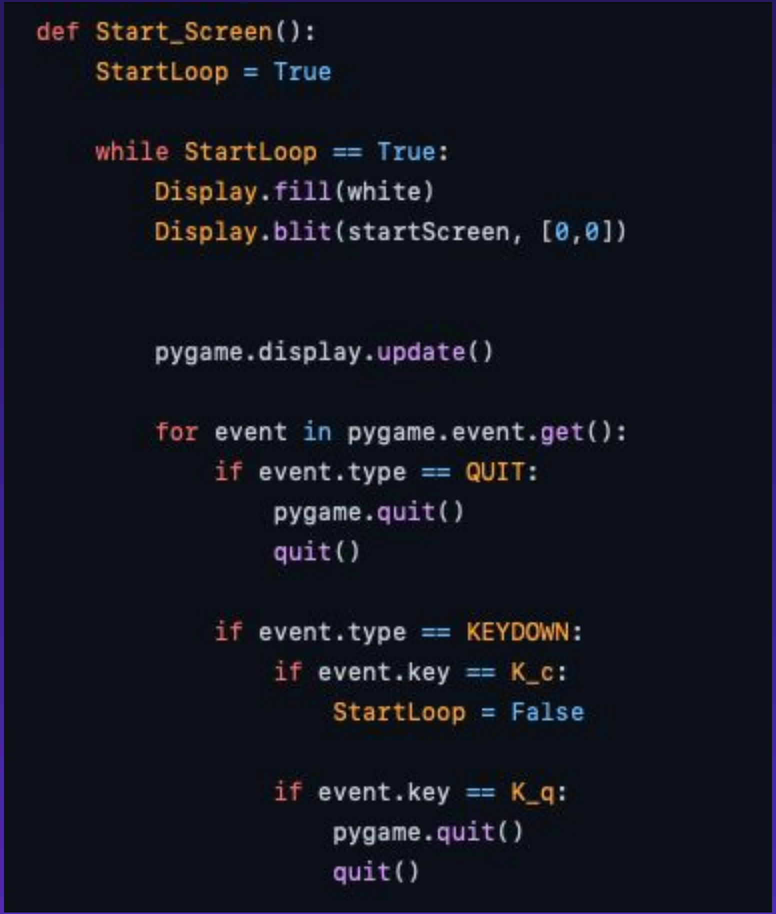
```python
mixer.pre_init(44100, -16, 2, 512)

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

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

## def Pause_Screen( )



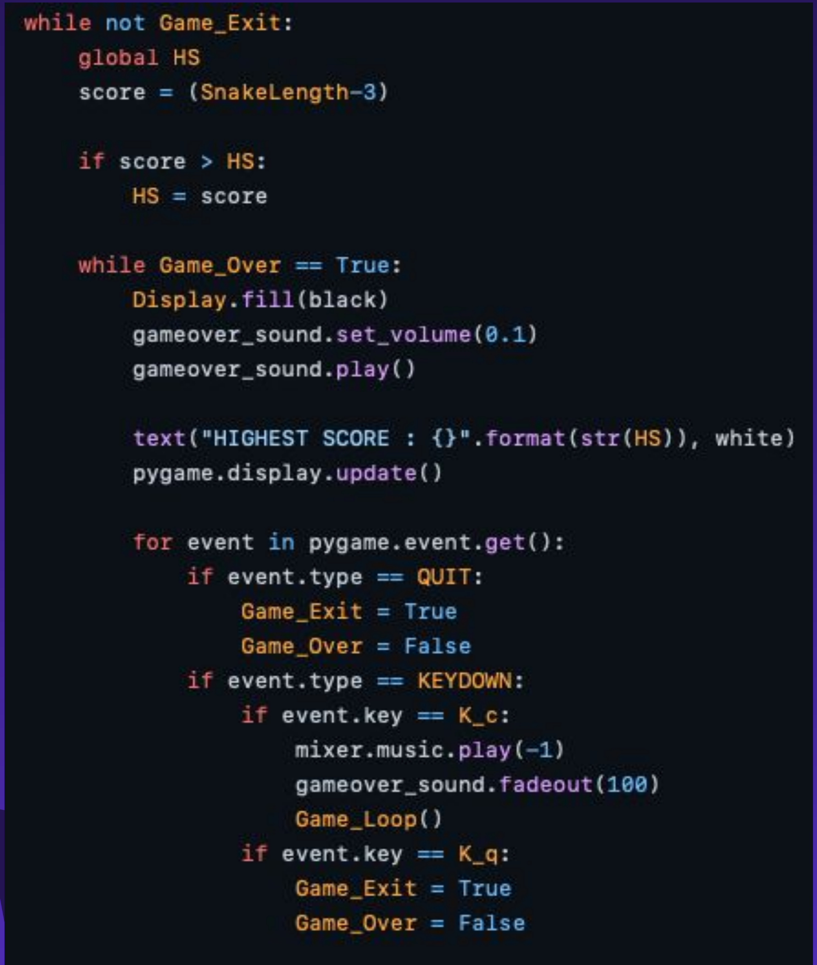
```python
def Pause_Screen():
    Pause = True
    mixer.music.fadeout(1000)
    while Pause:

        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)
```

## def Start_Screen( )

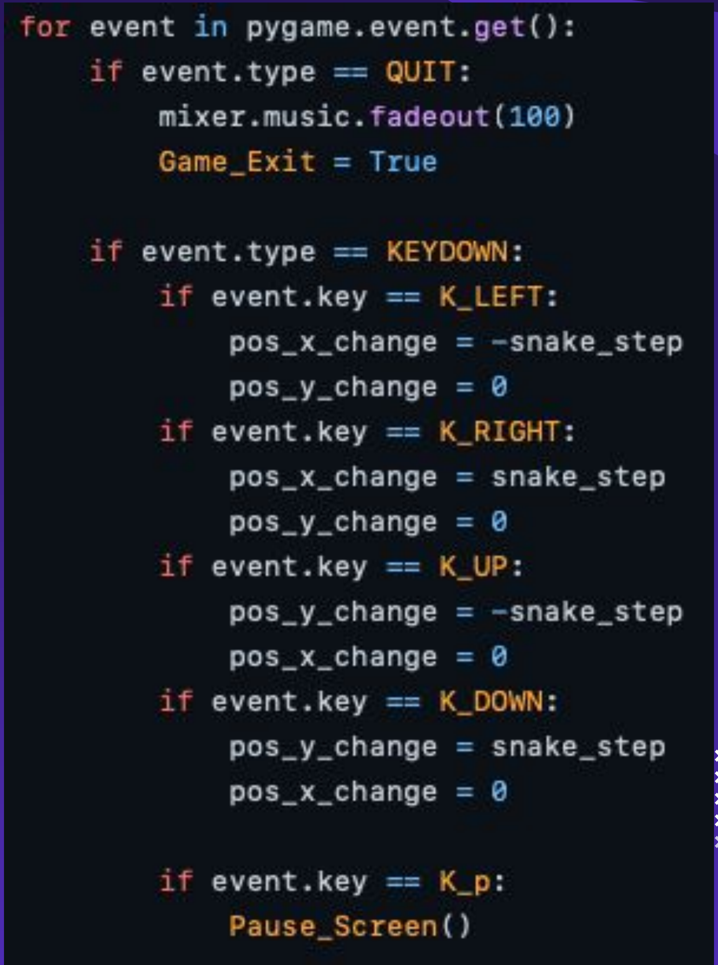


```
def Start_Screen():
    StartLoop = True

    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()
```

## def Game_Loop()

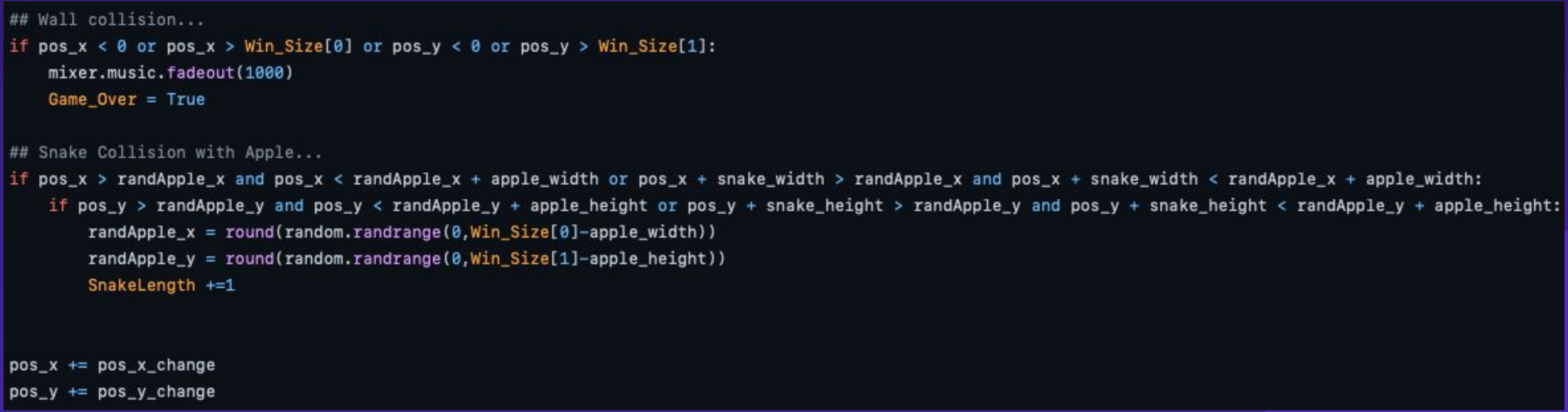


```
while not Game_Exit:
    global HS
    score = (SnakeLength-3)

    if score > HS:
        HS = score

    while Game_Over == True:
        Display.fill(black)
        gameover_sound.set_volume(0.1)
        gameover_sound.play()

        text("HIGHEST SCORE : {}".format(str(HS)), white)
        pygame.display.update()

        for event in pygame.event.get():
            if event.type == QUIT:
                Game_Exit = True
                Game_Over = False
            if event.type == KEYDOWN:
                if event.key == K_c:
                    mixer.music.play(-1)
                    gameover_sound.fadeout(100)
                    Game_Loop()
                if event.key == K_q:
                    Game_Exit = True
                    Game_Over = False
```



```
for event in pygame.event.get():
    if event.type == QUIT:
        mixer.music.fadeout(100)
        Game_Exit = True

    if event.type == KEYDOWN:
        if event.key == K_LEFT:
            pos_x_change = -snake_step
            pos_y_change = 0
        if event.key == K_RIGHT:
            pos_x_change = snake_step
            pos_y_change = 0
        if event.key == K_UP:
            pos_y_change = -snake_step
            pos_x_change = 0
        if event.key == K_DOWN:
            pos_y_change = snake_step
            pos_x_change = 0

        if event.key == K_p:
            Pause_Screen()
```

## Continuation:



```
## Wall collision...
if pos_x < 0 or pos_x > Win_Size[0] or pos_y < 0 or pos_y > Win_Size[1]:
    mixer.music.fadeout(1000)
    Game_Over = True

## Snake Collision with Apple...
if pos_x > randApple_x and pos_x < randApple_x + apple_width or pos_x + snake_width > randApple_x and pos_x + snake_width < randApple_x + apple_width:
    if pos_y > randApple_y and pos_y < randApple_y + apple_height or pos_y + snake_height > randApple_y and pos_y + snake_height < randApple_y + apple_height:
        randApple_x = round(random.randrange(0,Win_Size[0]-apple_width))
        randApple_y = round(random.randrange(0,Win_Size[1]-apple_height))
        SnakeLength +=1


pos_x += pos_x_change
pos_y += pos_y_change
```

# Thanks!

Do you have any questions?