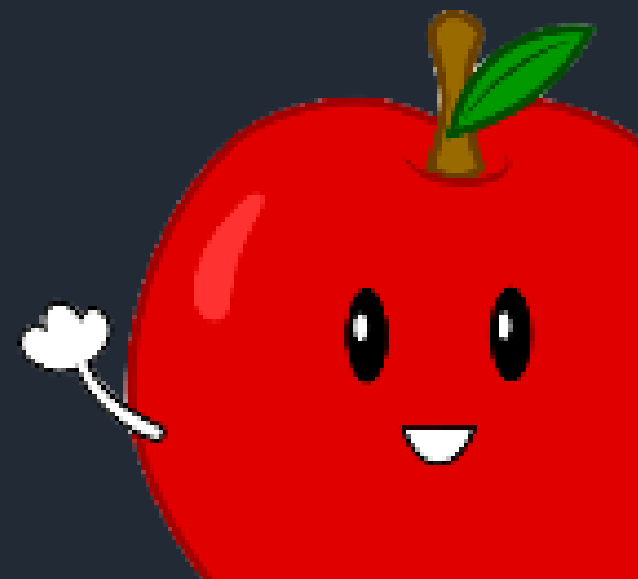
FAT WORM

By the group

WE NEED CACHE

VANESSA HOW 517120990081
KARINA LIANG 517120990057
JINWOO JEONG 517120990035
HYUNTAE JANG 517120990075



Abstract

Inspired by our childhood game,
SNAKE GAME

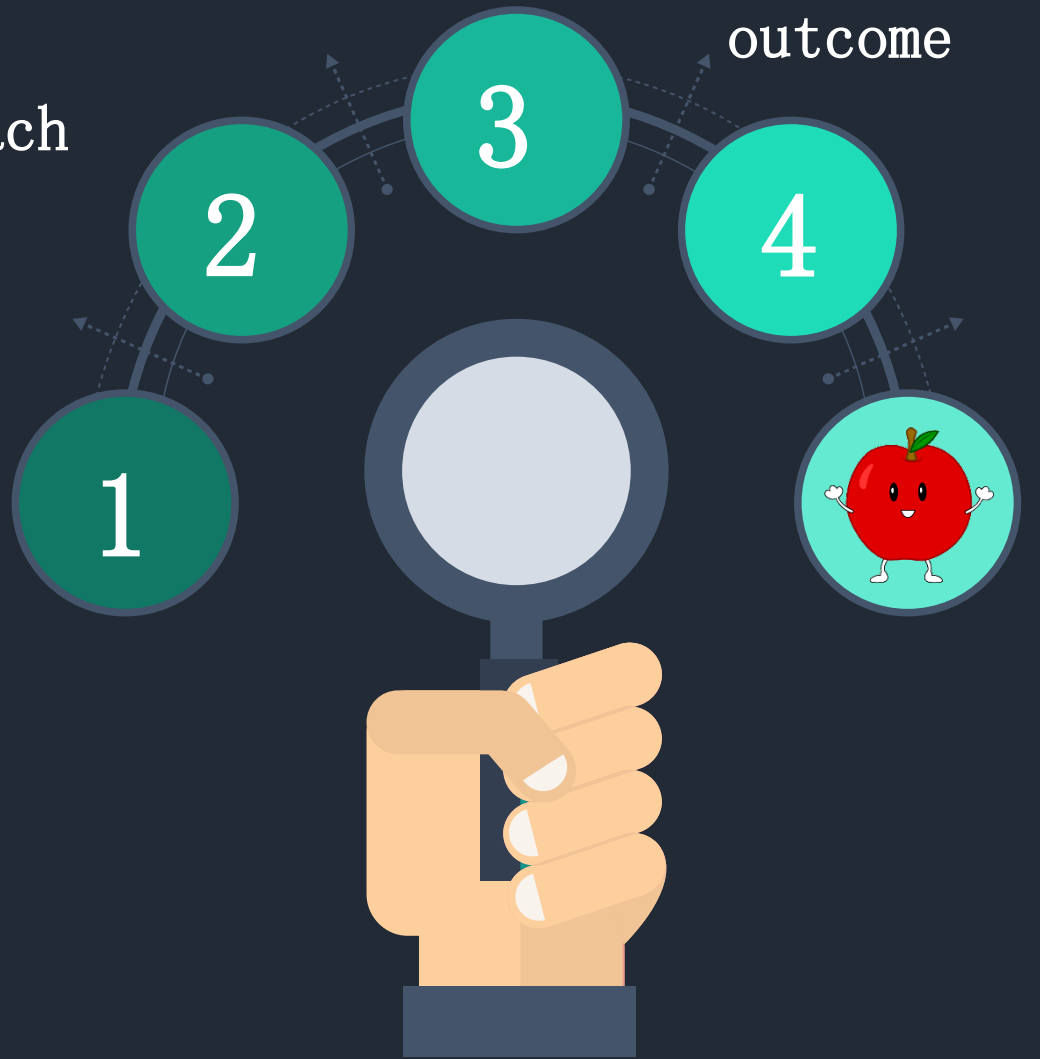
We created
OUR OWN!

problem

approach

coding
process

outcome



Introduction

Our Project “Fat Worm”

Is based on the “Snake Game”.

What is Snake Game?

A video game concept where the player maneuvers a line which grows in length, with the line itself being a primary obstacle.

From Blockade to…

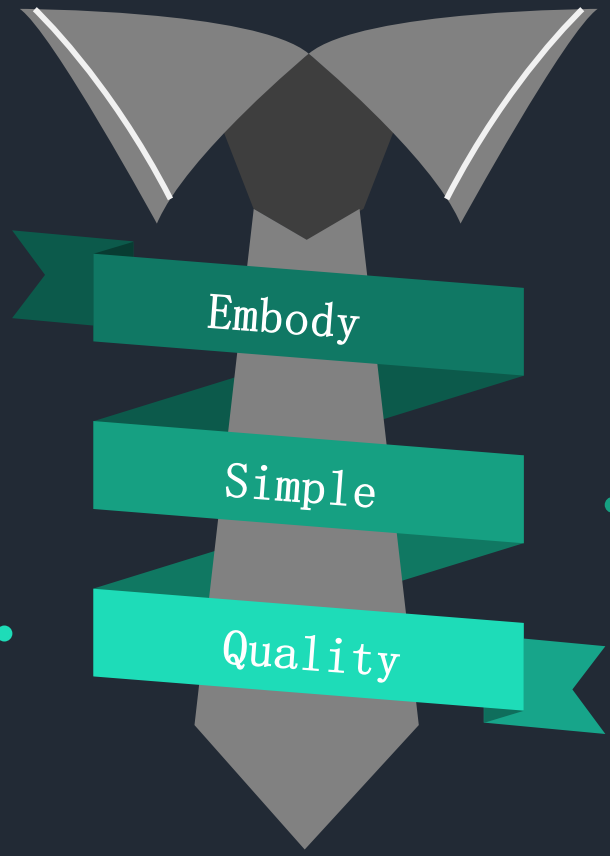
Originated in 1976 from the arcade game Blockade. Ease of implementing Snake resulted in numerous versions.

Nokia’s mobile phones

After a variant was preloaded on Nokia’s mobile phones in 1998, many people have become interested . More than 300 snake games in ios!!!

Objective

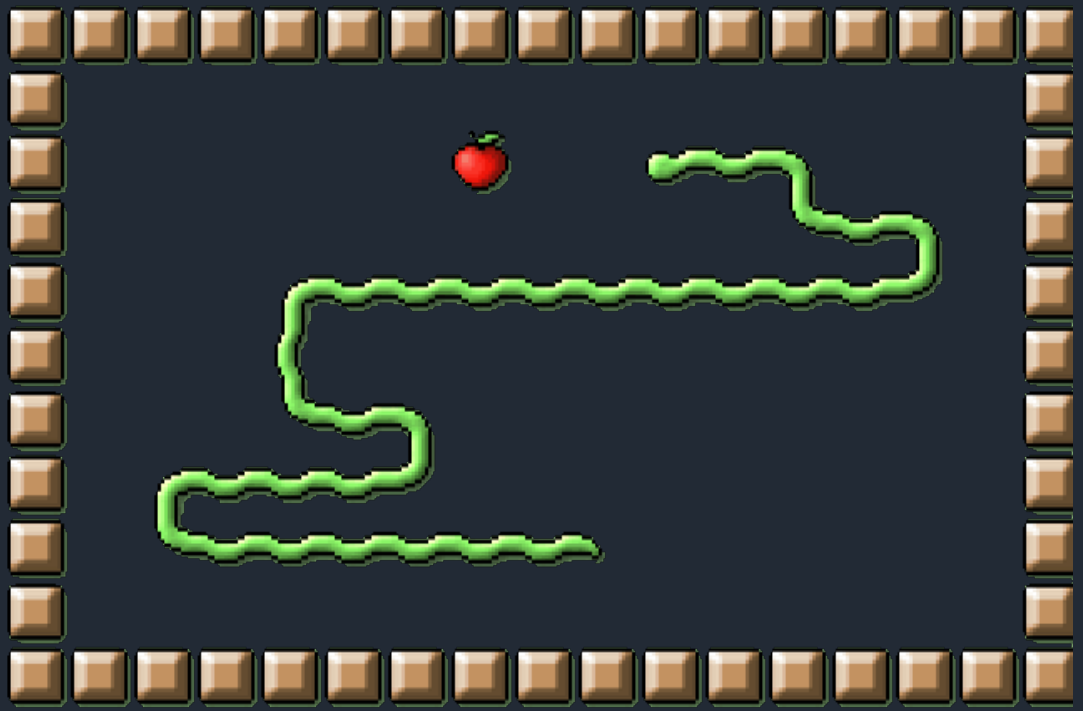
Embody
the whole “Snake Game”



Simplest codes and
methods of Python

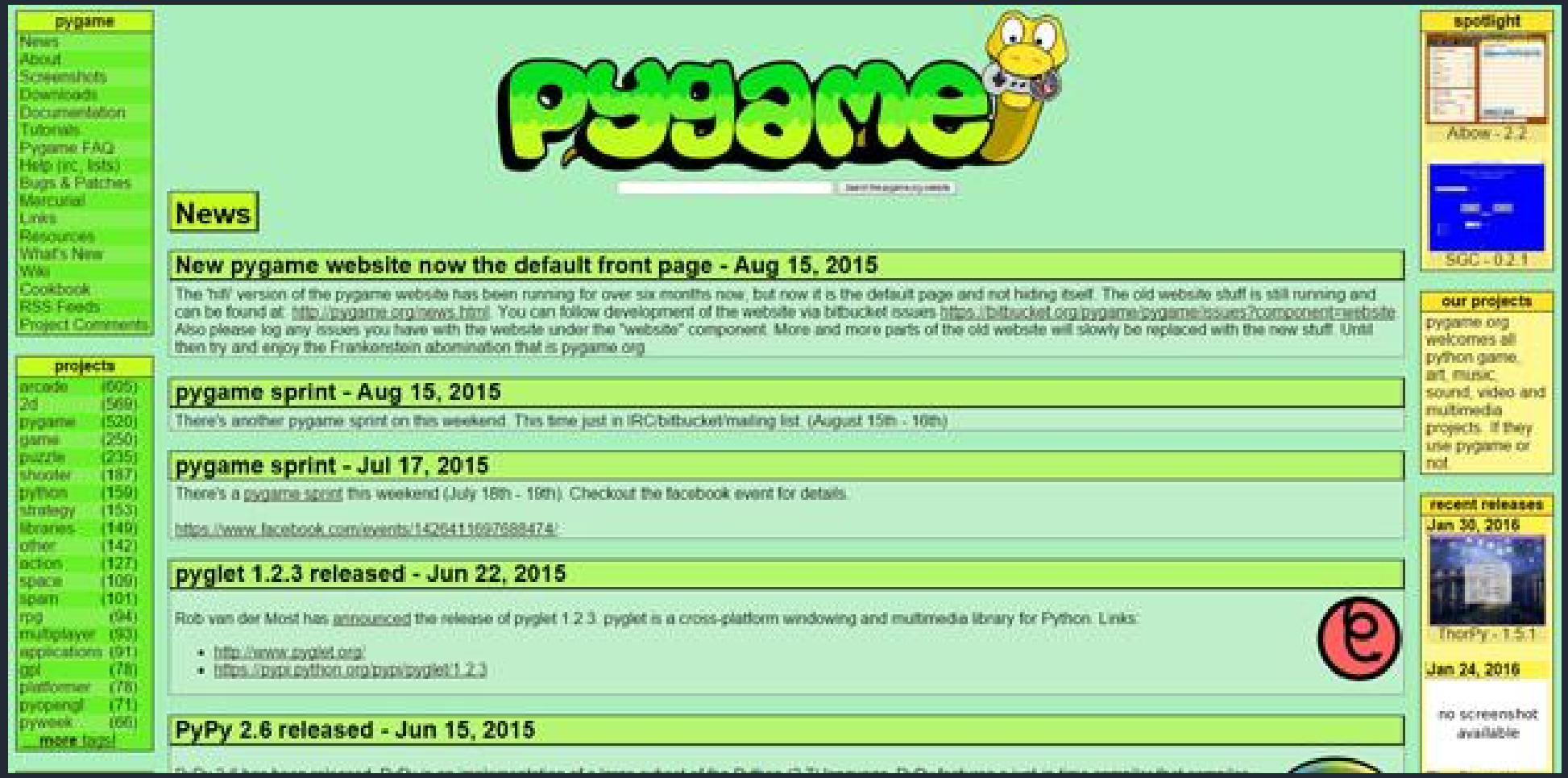
Make a High Quality game
that can be enjoyed by public

Game Play



1. Control dot, square or object on the plane
2. Snake leaves trail behind as it moves
3. Snake gets longer
4. Touches border or hits its body itself
5. GAME ENDS

Data Collection



Data Collection

Frets on Fire

Dangerous High School Girls in Trouble

“FAT WORM”

Pygame

- Cross-platform set of Python modules
- Written by Pete Shinners
- Community project
  under the open source free software

Built over the Simple DirectMedia Layer (SDL) library
Allowing real-time computer game development without the low-level mechanics of the C programming language and its derivatives

The most expensive functions inside games can be abstracted from the game logic!
Use a high-level programming language = Python, to structure the game

Coding Process

```
import pygame
import time
import random
```

Coding Process

❖The colors that you will use for your game

❖The font and color of the texts that will appear

Colors

Texts

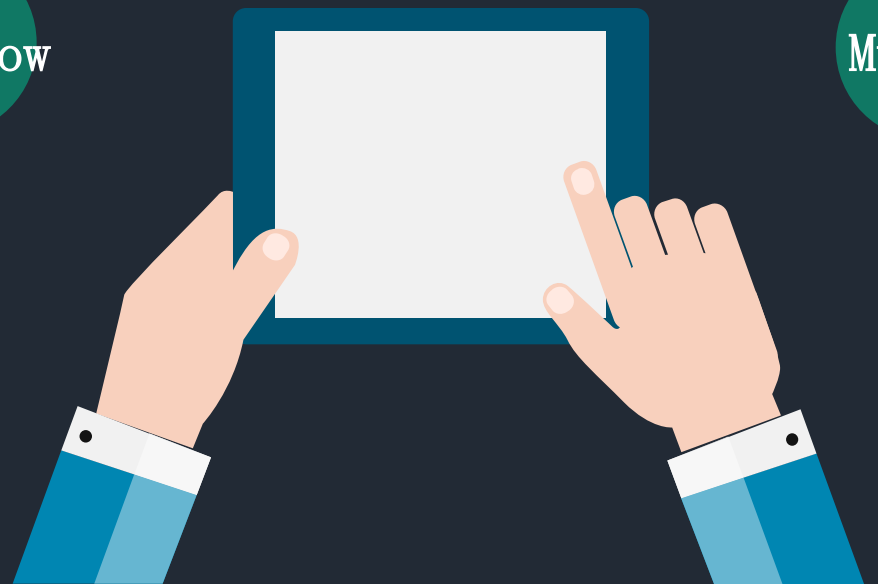
Window

Music

❖How big the display window will be

❖The name on top of the window

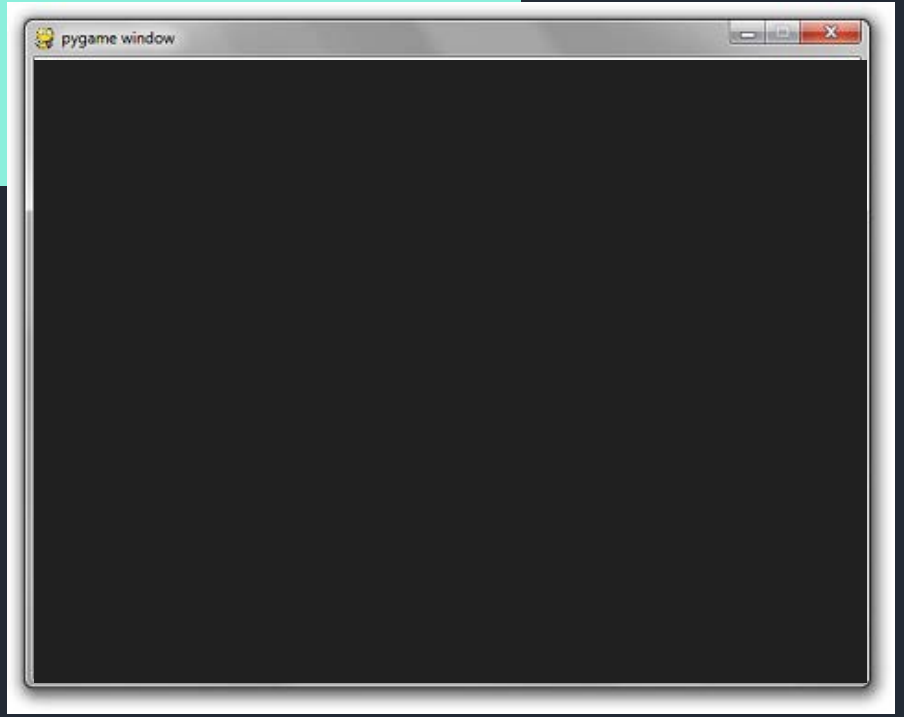
❖Sound effects and music throughout the game!



Window display

```
display_width = 800
display_height = 600
size = display_width, display_height
gameDisplay = pygame.display.set_mode(size)
pygame.display.set_caption("Worm Game!”)

icon = pygame.image.load("icon.png")
pygame.display.set_icon(icon)
```



Colors

```
white = (255, 255, 255)
black = (0, 0, 0)
red = (255, 0, 0)
green = (0, 155, 0)
purple = (255, 10, 255)
```

Texts

The code for the font, size and color!

```
smallfont = pygame.font.SysFont("comicsansms", 20)
medfont = pygame.font.SysFont("times", 35)
largefont = pygame.font.SysFont("宋体", 60)

def text_objects(text, color, size):
    if size == "small":
        textSurface = smallfont.render(text, True, color)
    elif size == "medium":
        textSurface = medfont.render(text, True, color)
    elif size == "large":
        textSurface = largefont.render(text, True, color)
    return textSurface, textSurface.get_rect()
```

This defines how a text message will show on the screen

```
def message_to_screen(msg, color, y_displace=0, size="small"):
    textSurf, textRect = text_objects(msg, color, size)
    textRect.center = (display_width / 2), (display_height / 2) + y_displace
    gameDisplay.blit(textSurf, textRect)
```

Features of the Worm Game

Bob the Worm

The player controls the worm's head

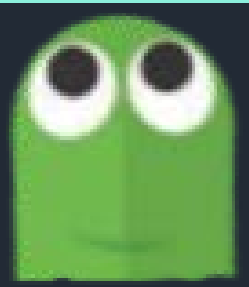
Apple

Apples that will randomly appear on the screen

Bob the Worm

Load the desired image, must be png format!

```
img = pygame.image.load("wormhead.png")
```



The code for how Bob's body will move

```
        if lead_x > randApplex and lead_x < randApplex + AppleThickness or
lead_x + block_size > randApplex and lead_x + block_size < randApplex +
AppleThickness:
        if lead_y > randAppley and lead_y < randAppley + AppleThickness:
            randApplex, randAppley = randAppleGen()
            snakeLength += 1
        elif lead_y + block_size > randAppley and lead_y + block_size <
randAppley + AppleThickness:
            randApplex, randAppley = randAppleGen()
            snakeLength += 1
```

Apple

Load the desired image, must be png format!

```
appleimg = pygame.image.load("apple.png")
```

The code for the apple to appear randomly

```
AppleThickness = 30
def randAppleGen():
    randApplex = round(random.randrange(0, display_width -
AppleThickness))
    randAppley = round(random.randrange(0, display_height -
AppleThickness))
    return randApplex, randAppley
```

Gameplay

❖How to play this game (how to control Bob)

❖How to pause and quit the game

❖Bob dies which means game over .

❖There are two situations in which Bob will die.

❖How to get score

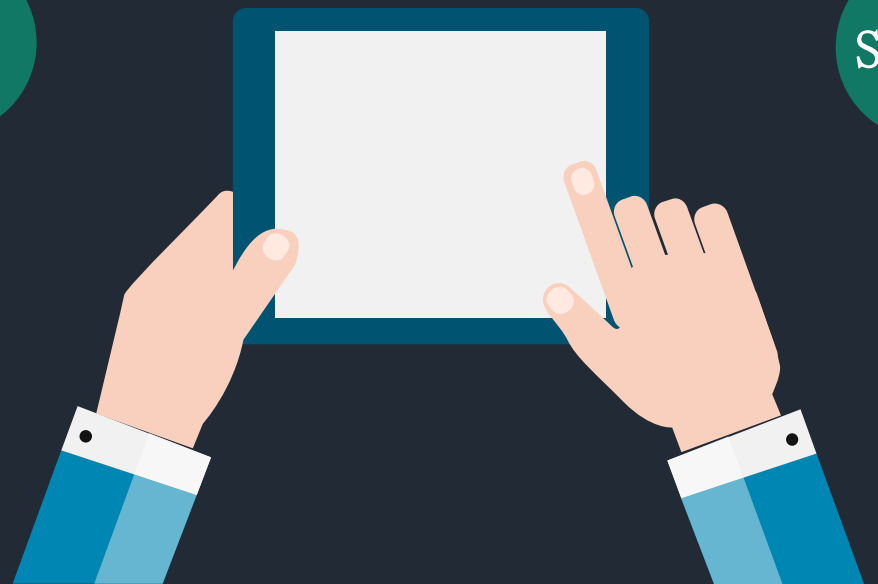
❖The score that you get will show on the screen.

Play

Pause & Quit

Game over

Score



Play

Firstly, some instructions for the player.

```
def game_intro():
    intro = True
    while intro:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                quit()
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_c:
                    intro = False
                if event.key == pygame.K_q:
                    pygame.quit()
                    quit()
        gameDisplay.fill(white)
        bg = pygame.image.load("introbackground.png")
        gameDisplay.blit(bg, [0,0])
        pygame.display.update()

        message_to_screen("Welcome to our Worm Game!", red, -100,"large")
        message_to_screen("As you know, the more apple you eat, the longer you become and the more scores you get!", black, -30)
        message_to_screen("But if you eat yourself or you run into the walls, you die!", black, 50)
        message_to_screen("Press C to play, P to pause, Q to quit.", black, 180)
        pygame.display.update()
        clock.tick(15)
```

Play

How to control Bob?

Player can press ‘up’, ’down’, ’left’, ’right’ to control Bob.

```
def snake(block_size, snakeList):
    if direction == "right":
        head = pygame.transform.rotate(img, 270)
    if direction == "left":
        head = pygame.transform.rotate(img, 90)
    if direction == "up":
        head = img
    if direction == "down":
        head = pygame.transform.rotate(img, 180)

    gameDisplay.blit(head, (snakeList[-1][0], snakeList[-1][1]))

    for XnY in snakeList[:-1]:
        pygame.draw.rect(gameDisplay, green, (XnY[0], XnY[1], block_size, block_size))
```

Pause & Quit

The player can press ‘P’ to pause the game if he wants to.

```
def pause():
    paused = True
    message_to_screen("Paused", white, -100, size="large")
    message_to_screen("Press C to continue or Q to quit", white, 40)
    pygame.display.update()
```

And while paused, player can press ‘C’ to continue the game or press ‘Q’ to quit the gam

```
while paused:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            quit()
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_c:
                paused = False
            elif event.key == pygame.K_q:
                pygame.quit()
                quit()
    clock.tick(5)
```

Game over

There are two situations in which Bob will die.
1). Bob dies if it crashes into the wall.

```
if lead_x >= display_width or lead_x < 0 or lead_y < 0 or lead_y >= display_height:
    gameOver = True
    dead_sound.play()
```

2). Bob dies if it eats itself.

```
if len(snakeList) > snakeLength:
    del snakeList[0]
for eachSegment in snakeList[:-1]:
    if eachSegment == snakeHead:
        gameOver = True
        dead_sound.play()
```

Game over

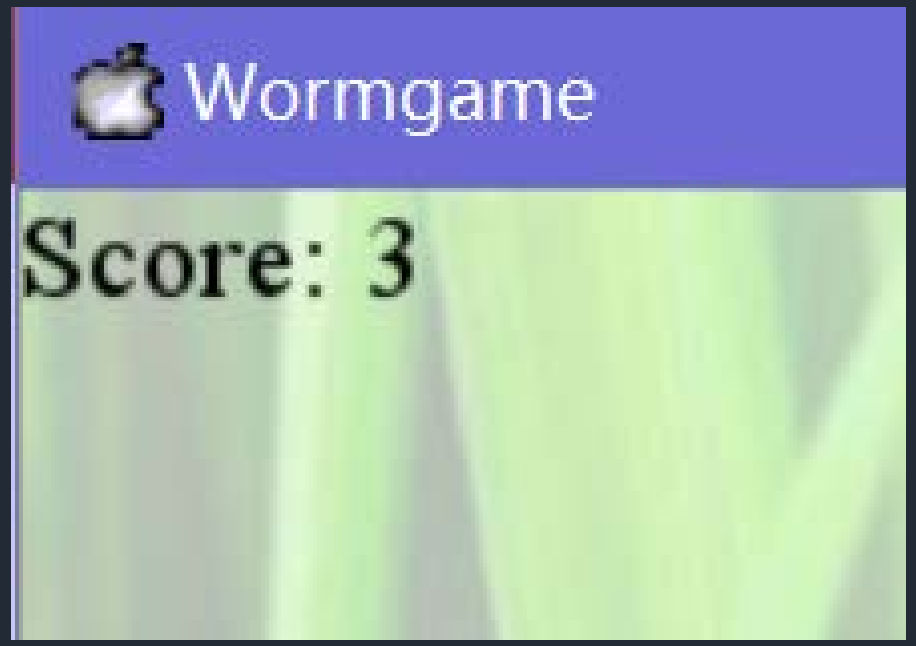
“gameOver” is False in the gameLoop, this is the lines for when gameOver is True, that is when Bob dies and it is Game Over.

```
while running:
    if gameOver == True:
        message_to_screen("Game over",red ,-50,size="large")
        message_to_screen("Press C to play again, or Q to quit", white, 50,
size="medium")
        pygame.display.update()
```

Score

Each time Bob eats one apple, the score increases by one.

```
def score(score):
    text = smallfont.render("Score: " + str(score), True, white)
    gameDisplay.blit(text, [0, 0])
```

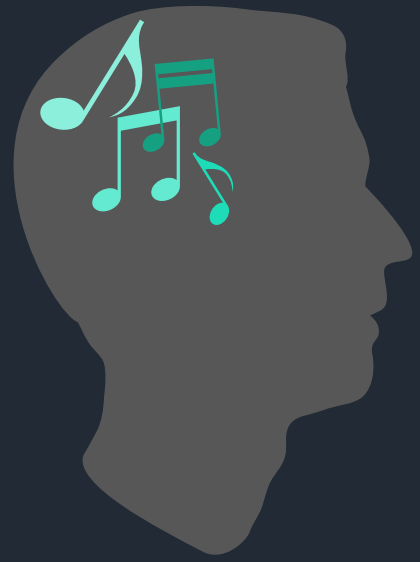


OPTIMIZATION

Background

```
gameDisplay.fill(white)
bg = pygame.image.load("background.png")
gameDisplay.blit(bg, [0,0])
pygame.display.update()
```

OPTIMIZATION



Music

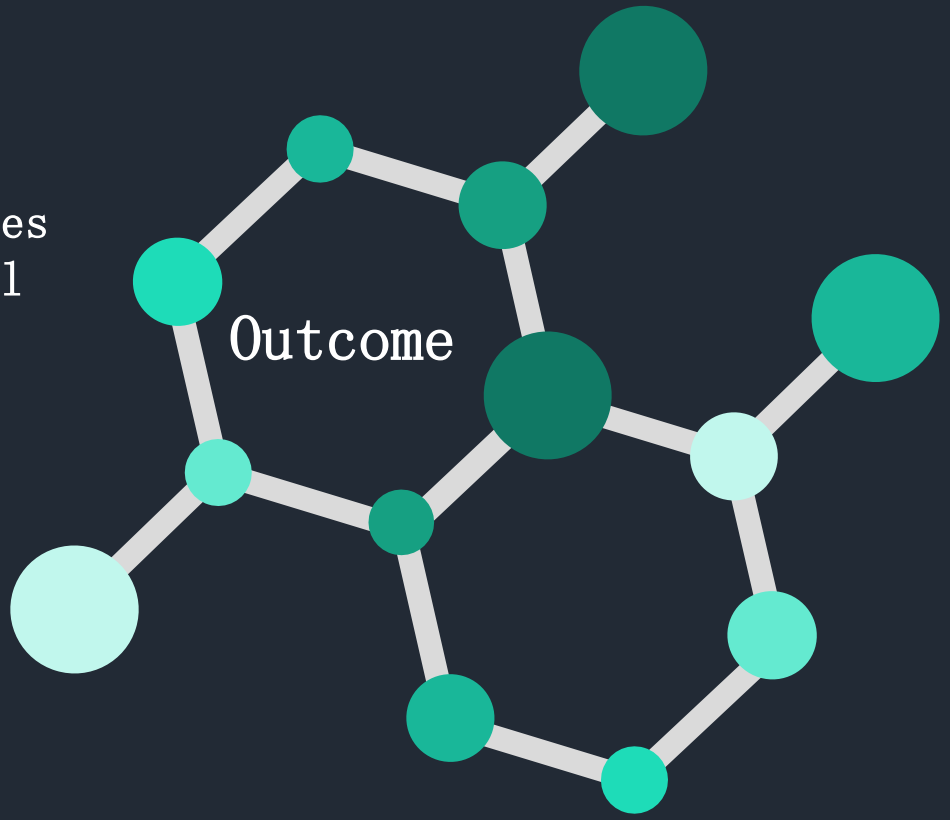
```
intro_sound = pygame.mixer.Sound(“intro.wav")

intro_sound.play()
```

Result

Programming requires math,
We gave a lot of tries to make it successful

Basic, good for beginners to understand



2-Dimensional
Not appealing

But we enjoyed this project,
Because we experienced every process

Challenges faced

Thank you Professor Baoyang!

01

Zero experience
in Pygame

02

Audio Problem
Why not working?
Audio should be .wav file

Thank You!