Algorithm:

1. Initialization:

Set initial values for game parameters:

minSpeed and maxSpeed for Fire objects.

Initial number of Fire objects.

Interval time to increase difficulty.

Score multiplier.

2. Game Loop:

While the game is running:

Update game elements (Gary's position, Fire's position, and possibly GaryFood's position).

Check for collisions between Gary and any Fire objects or GaryFood.

Increase score if Gary reaches GaryFood.

Reset GaryFood's position after it is collected.

End the game or reduce lives if Gary collides with Fire.

3. Difficulty Adjustment:

As time progresses, gradually increase:

The speed of Fire objects (minSpeed and maxSpeed).

The number of Fire objects, if applicable.

This can be managed through a timer that triggers every fixed interval (e.g., every 30 seconds).

4. Collision Detection:

Check if Gary's bounding box overlaps with any Fire objects or GaryFood.

Handle the outcomes:

If collide with Fire: Trigger game over or reduce Gary's health.

If collide with GaryFood: Increase the score and relocate GaryFood.

5. Score and Time Management:

Display and update the score based on game interactions.

Use a timer to manage game session length or survival time.

6. Game Over and Restart:

Provide options to restart the game or exit after game over.

Reset all parameters to their initial values upon restart.

import pygame, simpleGE, random

**Overview:**

Save Gary from the fire

class food as simpleGE.Sprite

define init as self and scene

Super() init scene

Self setImage as ("fire.png")

Self setSize as (35, 35)

Self minSpeed = 3

Self maxSpeed = 4

Self reset()

Define reset gets(self):

move to the top of screen

self.y = 10

x is random 0 - screen width

self.x = random.randint gets (0, self.screenWidth)

dy is random minSpeed to maxSpeed

self.dy = random.randint gets(self.minSpeed, self.maxSpeed)

def checkBounds(self):

if self.bottom > self.screenHeight:

self.reset()

class GaryFood as (simpleGE.Sprite):

Define init gets (self, scene):

Super init gets (scene)

Self set Image as("gary\_food.png")

Self set Size as (50, 50)

Self position = (520, 330)

class GaryFood as (simpleGE.Sprite):

Define init gets (self, scene):

Super init gets (scene)

Self set Image as("gary.png")

Self set Size as (50, 50)

Self position = (320, 400)

Self moveSpeed = 5

define process as(self):

if self isKeyPressed gets(pygame.K\_LEFT):

self.x = self move Speed

if self isKeyPressed as(pygame.K\_RIGHT):

self.x += self moveSpeed

if self.isKeyPressed(pygame.K\_UP):

self.y -= self.moveSpeed

Create class LblScore (simpleGE.Label):

define \_\_init\_\_(self):

super().\_\_init\_\_()

self.text = "Score: 0"

self.center = (100, 30)

Create class LblTime (simpleGE.Label):

define \_\_init\_\_(self):

super().\_\_init\_\_()

self.text = "Time: 10"

self.center = (500, 30)

class Game gets(simpleGE.Scene):

Define init gets(self):

Super init()

Self set Image as ("spongebac.jpeg")

Self sndFire = simpleGE.Sound("sound.mp3")

Self sndFire.play()

self.timer = simpleGE.Timer()

self.timer.start()

self.timerScore = simpleGE.Timer()

self.timerScore.start()

self.score = 0

self.lblTime =LblTime()

self.lblScore = LblScore()

self.Gary = Gary(self)

self.Fire = []

self.GaryFood = GaryFood(self)

for i in range(10):

self.Fire.append(Fire(self))

self.lblScore = LblScore()

self.lblTime = LblTime()

self.sprites = [self.Gary,

self.GaryFood,

self.Fire,

self.lblScore,

self.lblTime]

def process(self):

def process(self):

if self.timerScore.getElapsedTime() >= 2:

self.score += 3

#self.maxSpeed += 1

self.timerScore.start()

self.lblScore.text = f"Score: {self.score}"

if self.Gary.collidesWith(self.GaryFood):

print("Collision with GaryFood detected!")

#self.sndFire.play()

self.GaryFood.reset()

#self.lblScore.text = "Game Passed"

self.stop() # Handling after collision

for Fire in self.Fire:

if self.Gary.collidesWith(Fire):

print("yes")

Fire.reset()

self.stop()

#self.score += 1

self.lblTime.text = f"Time: {self.timer.getElapsedTime():.2f}"

class Instructions(simpleGE.Scene):

define \_\_init\_\_(self, score):

super().\_\_init\_\_()

self.setImage("spongebac.jpeg")

self.response = "Play"

self.instructions = simpleGE.MultiLabel()

self.instructions.textLines = [

"You are a gary trying to escape the fire destroying",

"the bikini bottom and also the annoying sound",

"Move with the left and right arrow keys",

"Your scores increases based on how long you",

"spend on the game",

"Have fun!"]

self.instructions.center = (320, 240)

self.instructions.size = (500, 250)

self.prevScore = score

self.lblScore = simpleGE.Label()

self.lblScore.text = f"Last score: {self.prevScore}"

self.lblScore.center = (320, 400)

self.btnPlay = simpleGE.Button()

self.btnPlay.text = "Play (up)"

self.btnPlay.center = (100, 400)

self.btnQuit = simpleGE.Button()

self.btnQuit.text = "Quit (down)"

self.btnQuit.center = (550, 400)

self.sprites = [self.instructions,

self.lblScore,

self.btnQuit,

self.btnPlay]

define process(self):

#buttons

if self.btnQuit.clicked:

self.response = "Quit"

self.stop()

if self.btnPlay.clicked:

self.response = "Play"

self.stop()

#arrow keys

if self.isKeyPressed(pygame.K\_UP):

self.response = "Play"

self.stop()

if self.isKeyPressed(pygame.K\_DOWN):

self.response = "Quit"

self.stop()

Define main():

keepGoing = True

score = 0

while keepGoing:

instructions = Instructions(score)

instructions.start()

if instructions.response == "Play":

game = Game()

game.start()

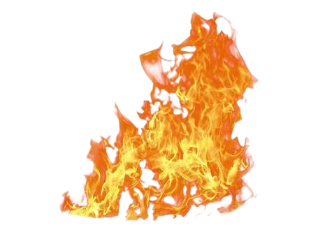
score = game.score

else:

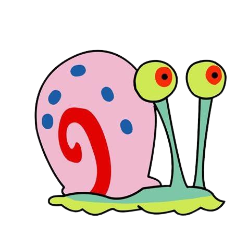
keepGoing = False

if \_\_name\_\_ =="\_\_main\_\_":

main



fire.png



gary.png



gary\_food.png



spongebac.png