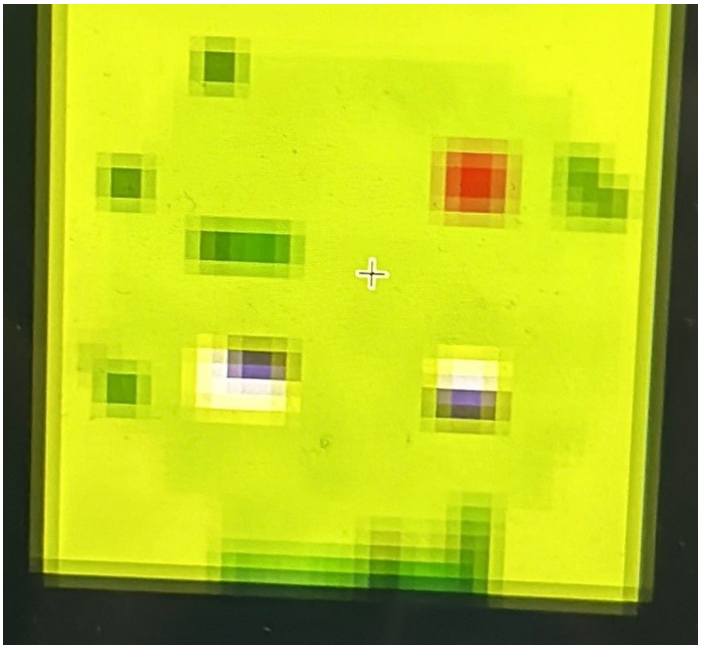
ZOMBIE BRAINS

Zombie Brains è un gioco che ho sviluppato nel linguaggio di programmazione Python: ho scelto questo linguaggio di programmazione perché lo avevo usato in precedenza per ricostruire un gioco già conosciuto “Snake”. Ho preferito riusare Python perché rispetto ad altri tipi di linguaggio come ad esempio CPP riesce con meno istruzioni a ottenere il risultato anche se la compilazione è più lenta.

Il progetto è partito in maniera un po’ casuale, quando stavo provando a caricare una grafica fatta a mano su un foglio di carta nel gioco snake citato prima, quando ho caricato il disegno nel progetto ho notato che la testa del serpente, per come l’avevo disegnata, sembrava lontanamente uno zombie...



Da lì è partita l’idea di Zombie Brain, e ho deciso di sviluppare completamente da solo il gioco. Mi sono inspirato per la difficoltà di girare tra i muri al famoso Pac-Man con la sola differenza che se prendi i muri perdi.

IL CODICE:

prima di tutto ho utilizzato la libreria `pygame` che mi ha consentito di utilizzare il lato grafico e non limitarmi al terminale e poi ho utilizzato anche la libreria `random` che mi ha consentito di generare i muri e i cervelli casualmente dentro la finestra

```
import pygame
import random
```

Subito dopo l'importazione ho inizializzato la libreria `pygame`, la grandezza della finestra, e le texture dei blocchi usati e la loro dimensione. Ho deciso di dichiarare più texture per lo stesso zombie, per il quale ho fatto una texture per ogni lato in cui può girare. Approfondiremo più tardi come ho risolto questo problema.

```
# Inizializza Pygame
pygame.init()

# Imposta le dimensioni della finestra
window_width = 800
window_height = 600

window = pygame.display.set_mode((window_width, window_height
pygame.display.set_caption("Zombie Brains")

# Definisci i colori
black = (0, 0, 0)
green = (0, 255, 0)


# Carica le immagini
zombie_head_up = pygame.image.load("block_part/zombie_head_up
zombie_head_up = pygame.transform.scale(zombie_head_up, (20,
zombie_head_down = pygame.image.load("block_part/zombie_head_
zombie_head_down = pygame.transform.scale(zombie_head_down, (
zombie_head_left = pygame.image.load("block_part/zombie_head_
zombie_head_left = pygame.transform.scale(zombie_head_left, (
zombie_head_right = pygame.image.load("block_part/zombie_head
zombie_head_right = pygame.transform.scale(zombie_head_right,
brain = pygame.image.load("block_part/brain.png")
brain = pygame.transform.scale(brain, (20, 20))
wall = pygame.image.load("block_part/wall.png")
wall = pygame.transform.scale(wall, (20, 20))
```

```
# Definisci le dimensioni dello zombie, del muro e del cervel
zombie_size = 20
brain_size = 20
wall_size = 20
```

Qui ho fatto un set up iniziale per far partire il gioco quindi ho prima dato le coordinate iniziali dello zombie, poi ho inizializzato a 0 lo spostamento dello zombie e ho generato delle coordinate casuali per il cervello e per il muro usando la libreria `random.` Le coordinate del muro sono state inserite in una tupla che verrà aggiornata col tempo per tenere traccia di tutti i muri presenti in mappa e non farli cancellare mano a mano; infine ho inizializzato il punteggio, la velocità che ho deciso di mantenere costante perché la difficoltà è più incentrata sullo schivare i muri e non su quanto vada veloce lo zombie.

```
# Definisci la posizione iniziale dello zombie
zombie_x = window_width // 2
zombie_y = window_height // 2

# Definisci il vettore di movimento iniziale
x_change = 0
y_change = 0


# Crea cervello e crea muro iniziale
brain_x = random.randrange(brain_size, window_width - brain_s
brain_y = random.randrange(brain_size, window_height - brain_
wall_x = random.randrange(wall_size, window_height - wall_siz
wall_y = random.randrange(wall_size, window_height - wall_siz

walls = [(wall_x, wall_y)]

# Inizializza il punteggio e la velocità e i cervelli presi
score = 0
speed = 8
brain_taken = 0
```

Ho creato questa funzione notando che nello sviluppo del codice e nei vari test ci poteva essere una probabilità che il cervello fosse stato generato nelle

stesse coordinate di un muro, quindi ho dichiarato questa funzione che innanzitutto genera casualmente le coordinate di un nuovo cervello poi finché la condizione è vera del: coordinate cervello = coordinate muro per ogni muro già generato e salvato nella tupla di cui parlvavo precedentemente, genera un'altro cervello finché non collide con nessun muro.

```
def generate_brain_position():
    brain_x = random.randrange(brain_size, window_width - bra
    brain_y = random.randrange(brain_size, window_height - br

    # Verificare se la posizione del cervello non collide con
    while any((brain_x == wall_x and brain_y == wall_y) for w
        brain_x = random.randrange(brain_size, window_width -
        brain_y = random.randrange(brain_size, window_height

    return brain_x, brain_y
```

Della funzione `draw_game` mi soffermerei sulla parte che decide quale posizione dello zombie disegnare: qui ho verificato se il cambiamento delle coordinate dello zombie fosse stato 0 per le x e in negativo per le y, lo zombie avrebbe guardato verso l'alto, se invece il cambiamento delle x fosse stato 0 e il cambiamento delle y in positivo avrebbe guardato verso il basso, se invece il cambiamento delle x fosse stato in negativo e il cambiamento delle y fosse stato invariato avrebbe guardato verso sinistra, e se il cambiamento delle x fosse stato positivo e il cambiamento delle y nullo avrebbe guardato verso destra. Ad ogni condizione ho assegnato una texture che ho inizializzato ad inizio codice.

```
    if x_change == 0 and y_change == -zombie_size:
        window.blit(zombie_head_up, (zombie_x, zombie_y))
    elif x_change == 0 and y_change == zombie_size:
        window.blit(zombie_head_down, (zombie_x, zombie_y))
    elif x_change == -zombie_size and y_change == 0:
        window.blit(zombie_head_left, (zombie_x, zombie_y))
    elif x_change == zombie_size and y_change == 0:
        window.blit(zombie_head_right, (zombie_x, zombie_y))
```

Una delle parti più importanti, nonché la base per far funzionare il gioco, è il game loop quindi ripetere le istruzioni ogni volta per far scorrere il gioco fluidamente.

Tratterò le parti più importanti del game loop:

questo pezzo del game loop verifica se l'utente ha innanzitutto premuto la tastiera e se la risposta è positiva, verifica se ha premuto una delle quattro freccettine. Se ha permuto per esempio quella di sinistra il cambiamento delle x andrà in negativo per la grandezza dello zombie e invece il cambiamento delle y rimarrà invariato. Questo verrà fatto per tutte le altre frecce con i loro rispettivi cambiamenti.

```
        elif event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                x_change = -zombie_size
                y_change = 0
            elif event.key == pygame.K_RIGHT:
                x_change = zombie_size
                y_change = 0
            elif event.key == pygame.K_UP:
                x_change = 0
                y_change = -zombie_size
            elif event.key == pygame.K_DOWN:
                x_change = 0
                y_change = zombie_size
```

In questa parte del game loop vengono sommati prima i cambiamenti suddetti alla posizione dello zombie, poi si verifica se lo zombie è uscito fuori dai limiti della finestra: in tal caso contrassegno il `game_over` come vero e si interrompe il gioco, in seguito si controlla se lo zombie collide con ogni muro presente in gioco in questa eventualità si interrompe il gioco anche qui. Dopo aver verificato quanto detto, si controlla se lo zombie ha preso un cervello e se le coordinate coincidono viene generato un'altro muro e poi si usa la funzione `generate_brain_position` creata in precedenza per generare un brain che non coincida con nessun muro. Infine si aggiunge alla tupla le coordinate del nuovo muro

```
    zombie_x += x_change
    zombie_y += y_change
```

```
    # Controlla le collisioni con i bordi della finestra
    if zombie_x < 0 or zombie_x >= window_width or zombie_y <
        game_over = True

    # Controlla le collisioni con il corpo dello zombie con i
    for wall_x, wall_y in walls:
        if zombie_x == wall_x and zombie_y == wall_y:
            game_over = True
            break

    # Controlla se lo zombie ha mangiato il cervello e genera
    if zombie_x == brain_x and zombie_y == brain_y:
        wall_x = random.randrange(wall_size, window_width - w
        wall_y = random.randrange(wall_size, window_height - 
        brain_x, brain_y = generate_brain_position()

        walls.append((wall_x, wall_y))
        # Aggiungi nuove coordinate del muro come tupla
```

Al termine ho deciso di trattare il punteggio in maniera esponenziale: più cervelli vengono presi, più muri ci saranno, più aumenterà la difficoltà. Quindi il punteggio crescerà di conseguenza.

```
brain_taken += 1

        if brain_taken < 10:
            score += 5
        elif brain_taken < 20:
            score += 15
        elif brain_taken < 40:
            score += 40
        elif brain_taken > 40:
            score += 50
```

Ritengo che questo programma sia il progetto maggiormente rappresentativo dei miei progressi e delle mie competenze sviluppate durante questo anno scolastico.