BÁO CÁO ĐỒ ÁN MÔN HỌC
Lớp: IE221.O21.CNCL

SINH VIÊN THỰC HIỆN:

Mã sinh viên: 21520746 **Họ và tên: Nguyễn Tuấn Dũng**

TÊN ĐỀ TÀI: SIMPLE GAME 2D

LINK GITHUB

DNoman/2D-GAME (github.com)

CÁC NỘI DUNG CẦN BÁO CÁO:

1. Giới thiệu đồ án

   _ Thực hiện tạo 1 trò chơi có nhân vật , map và 1 số tác nhân di chuyển xung quanh (AI đơn giản). Mục tiêu là tạo ra nhân vật cầm vũ khí tương tác với các tác nhân di chuyển xung quanh map và qua map mới (dự kiến 4 map). Nhân vật và các tác nhân đều có animation chuyển động, khi tương tác giữa nhân vật và tác nhân sẽ gây hiệu ứng mất máu. Nhân vật có thể tương tác với môi trường bằng cách nhặt các vật phẩm để hồi máu hoặc đồng xu tích điểm thưởng trong các màn của trò chơi và không thể đi qua được các bức tường đã cố định sẵn. Game sẽ có Over Screen khi trò chơi kết thúc hoặc nhân vật bị giết và sẽ có âm nhạc và hiệu ứng âm thanh khi nhân vật tương tác với môi trường xung quanh. Game sẽ có giao diện để người chơi tương tác với game (ví dụ bắt đầu trò chơi, thoát trò chơi,…)

2. Quá trình thực hiện

   a. Tuần 1: Xây dựng nhân vật, Animate nhân vật, Animate goblin và weapon + Set up nhân vật bằng các hình ảnh cắt từ ASSET image để tạo thành hiệu ứng animation đứng yên và chạy (8 image , 4 image tạo hiệu ứng animate cho đứng yên và 4 image tạo hiệu ứng animate khi di chuyển)

+ Set up goblin bằng các hình ảnh cắt từ ASSET image để tạo thành hiệu ứng animation đứng yên và chạy (8 image , 4 image tạo hiệu ứng animate cho đứng yên và 4 image tạo hiệu ứng animate khi di chuyển)
+ Lắp thêm cho nhân vật cái cung. Tạo hiệu ứng cho cung chạy theo con trỏ chuột.
+ Lắp điều khiển nhân vật di chuyển bằng các nút AWSD
+Lắp mũi tên cho nhân vật khi kích chuột và mũi tên bay ra theo chiều con trỏ chuột. Tạo them sát thương cho mũi tên khi dính vào đối tượng
+ Tạo cho goblin thanh máu và tương tác giữa goblin và mũi tên . Nếu mũi tên dính goblin thì goblin sẽ bị mất máu do mũi tên mang lại.

b. Tuần 2: Tạo ra hoạt ảnh và Map

+ Tạo hoạt ảnh cho kẻ thù khi bị đánh bằng cách đánh vào sẽ hiển thị lượng máu bị trừ phía trên đầu
+ Hiện thị lượng máu nhân vật đang điều khiển bằng cách thêm các hoạt ảnh trái tim bên trên màn hình, nếu bị đánh sẽ trừ máu (ví dụ mất trái tim)
+ Thêm trên map những bình máu và khi tiếp xúc với những bình máu sẽ hồi lại lượng máu (ví dụ hồi lại là 1 trái tim)
+ Tạo ra 1 map đơn giản bằng cách thêm từng phần tử pixel khối vào tạo thành 1 map lớn cho 1 màn chơi (Từng pixel có thể kiếm từ nguồn trên mạng)
+ Map rộng hơn so với khung hình pygame, thiết lập camera chạy theo nhân vật để khám Map.
+ Tương tác với map, ví dụ không thể đi xuyên qua được tường, mũi tên bắn không qua được tường ,…

c. Tuần 3: Tương tác với AI của từng đối tương

+ Tạo ra 3 màn chơi khác nhau có những đối tượng tương tác với player khác nhau.

+ Tạo ra AI cho từng đối tượng, sẽ có nhiều hơn 1 đối tượng kẻ thù là goblin , mỗi đối tượng sẽ có thao tác giống nhau ở tốc độ, sát thương (các AI này sẽ đánh tầm gần)

+ Tạo ra AI BOSS AI đánh tầm xa, sát thương và cách di chuyển khác những con AI bên trên.

d. Tuần 4: Hoàn thành tựa game

+ Tương tác với các màn chơi , lên các mức level cao hơn bằng cách nhặt xu hoặc đánh quái hoặc di chuyển tới địa điểm cần thiết để qua màn.

+ Thêm hiệu ứng mỗi khi qua màn mới bằng hoạt ảnh chuyển cảnh

+ Tạo ra nút ấn khi chết sẽ hiện ra chữ restart lại màn chơi

+ Tạo ra Game menu (ấn Start để bắt đầu trò chơi hoặc Exit để thoát trò chơi)

+ Thêm Sound effect vào trò chơi, ví dụ xuyên suốt trò chơi sẽ có nhạc, khi đánh kẻ thù hoặc nhặt vật phẩm sẽ có sound nhạc effect bật lên

3. Kết quả đạt được

Tuần 1: Xây dựng nhân vật, Animate nhân vật, Animate goblin và weapon

+ Set up nhân vật bằng các hình ảnh cắt từ ASSET image để tạo thành hiệu ứng animation đứng yên và chạy (8 image , 4 image tạo hiệu ứng animate cho đứng yên và 4 image tạo hiệu ứng animate khi di chuyển)

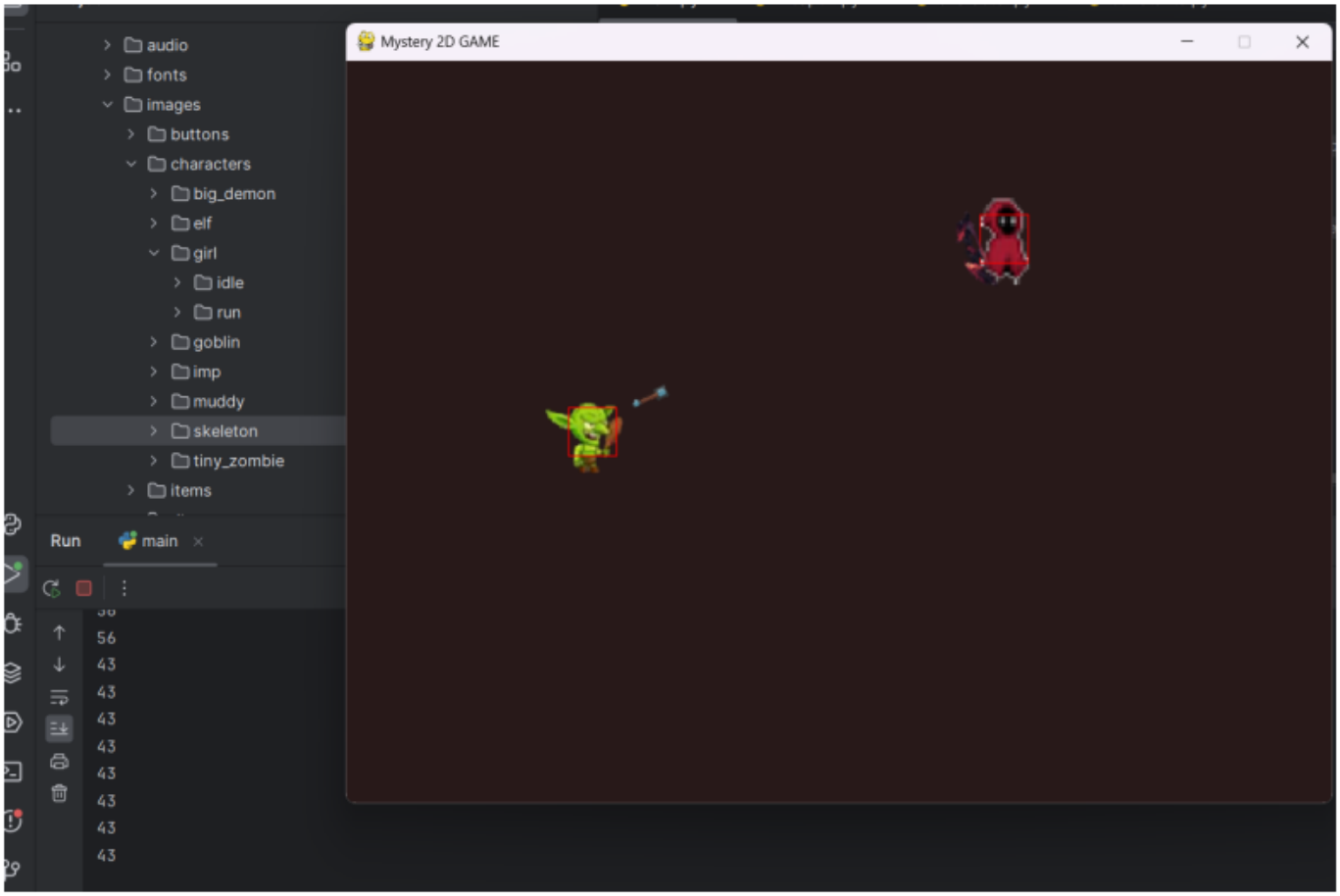
+ Set up goblin bằng các hình ảnh cắt từ ASSET image để tạo thành hiệu ứng animation đứng yên và chạy (8 image , 4 image tạo hiệu ứng animate cho đứng yên và 4 image tạo hiệu ứng animate khi di chuyển)

+ Lắp thêm cho nhân vật cái cung. Tạo hiệu ứng cho cung chạy theo con trỏ chuột.

+ Lắp điều khiển nhân vật di chuyển bằng các nút AWSD

+Lắp mũi tên cho nhân vật khi kích chuột và mũi tên bay ra theo chiều con trỏ chuột. Tạo them sát thương cho mũi tên khi dính vào đối tượng

+ Tạo cho goblin thanh máu và tương tác giữa goblin và mũi tên . Nếu mũi tên dính goblin thì goblin sẽ bị mất máu do mũi tên mang lại.



Tuần 2:

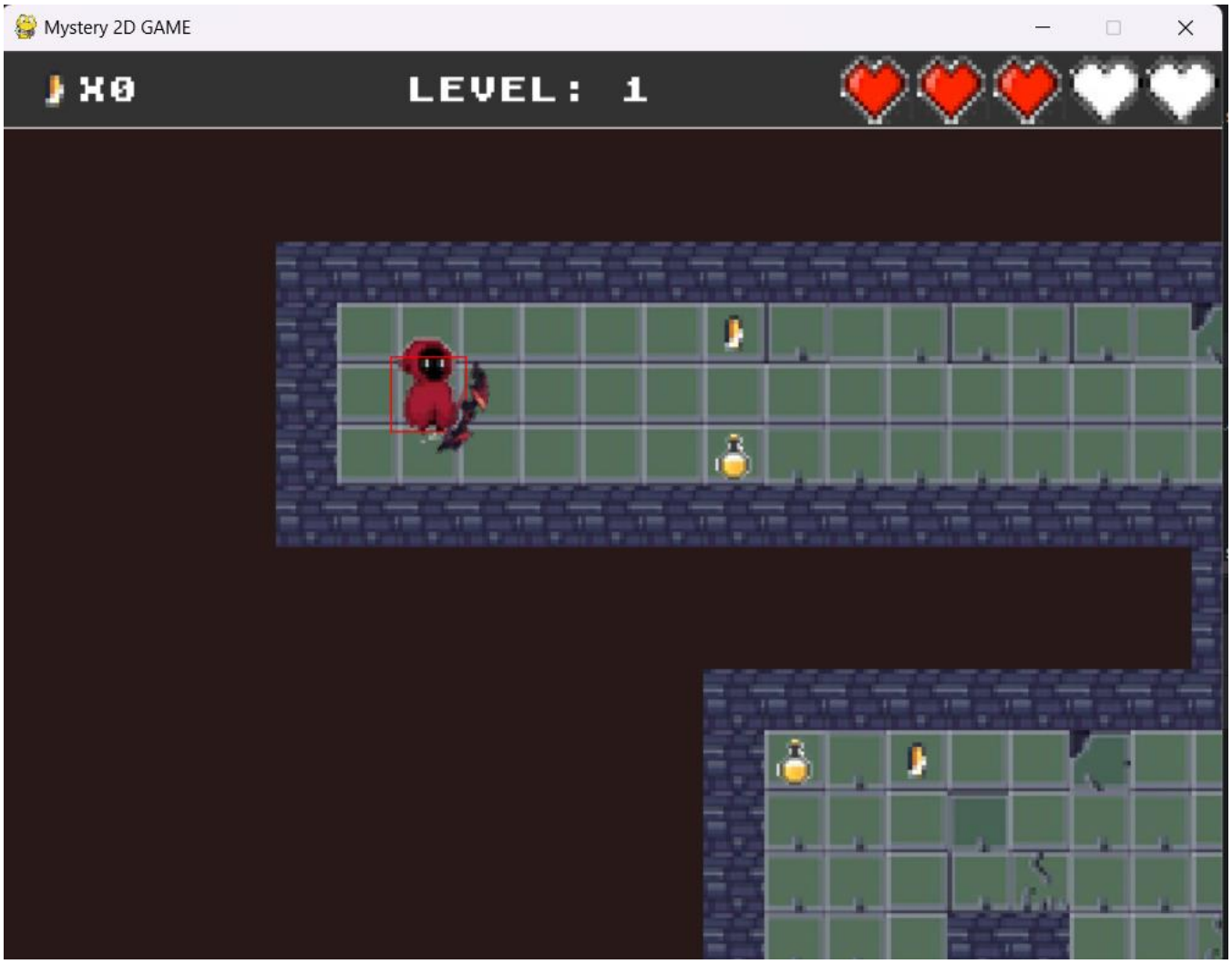
+ Thêm trên map những bình máu và khi tiếp xúc với những bình máu sẽ hồi lại lượng máu (ví dụ hồi lại là 1/2 trái tim)

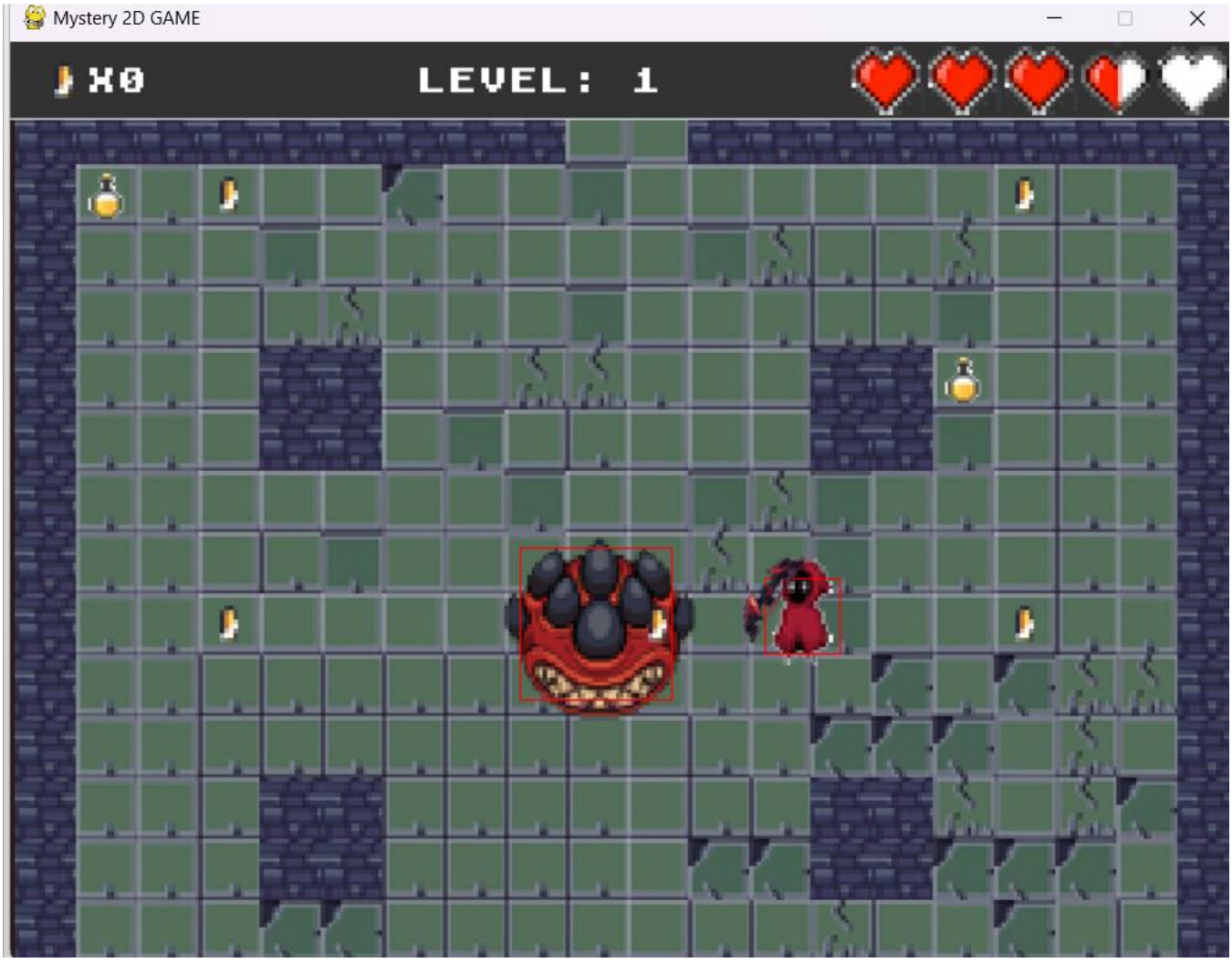
+ Tạo ra 1 map đơn giản bằng cách thêm từng phần tử pixel khối vào tạo thành 1 map lớn cho 1 màn chơi (Từng pixel có thể kiếm từ nguồn trên mạng)

+ Map rộng hơn so với khung hình pygame, thiết lập camera scroll chạy theo nhân vật để khám phá Map.

+ Hiện thị lượng máu nhân vật đang điều khiển bằng cách thêm các hoạt ảnh trái tim bên trên màn hình, nếu bị đánh sẽ trừ máu (ví dụ mất trái tim)

+Tạo file map bằng file csv đơn giản rồi đưa vào python ghép các pixel để tạo thành 1 map hoàn chỉnh


Mystery 2D GAME
X0
LEVEL: 1



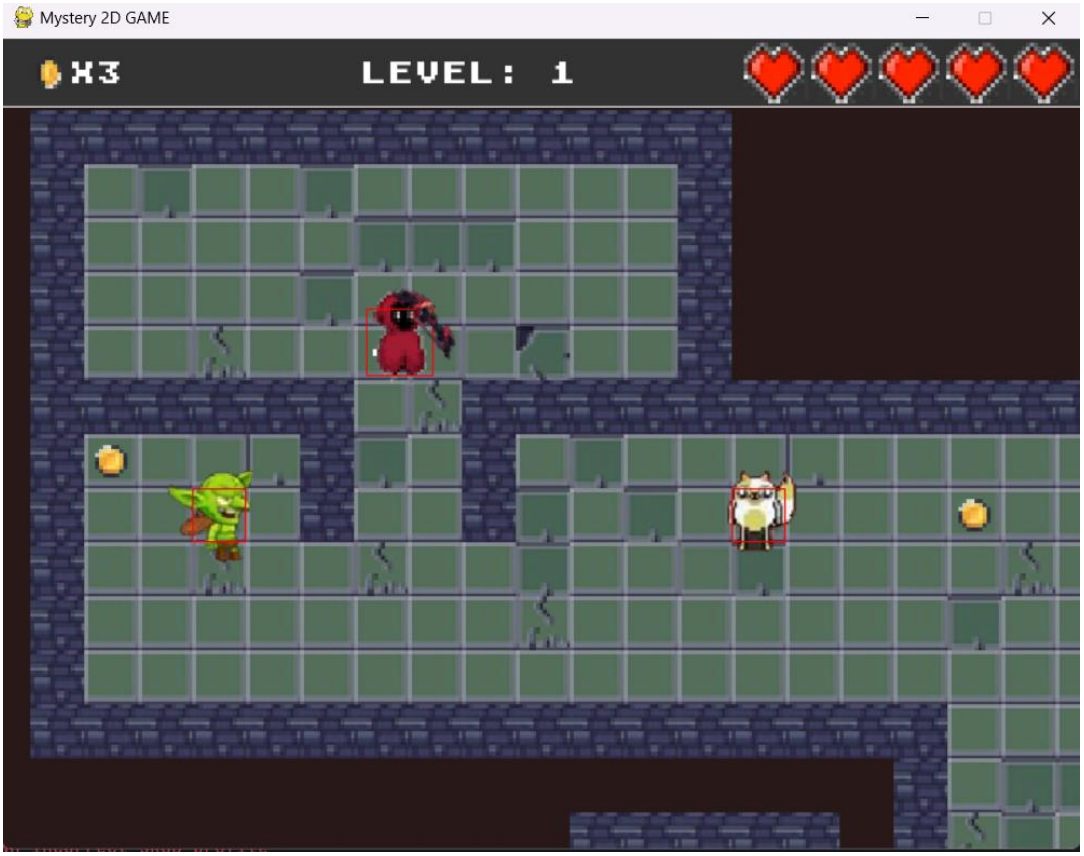
+ Tạo hoạt ảnh cho kẻ thù khi bị đánh bằng cách đánh vào sẽ hiển thị lượng máu bị trừ phía trên đầu

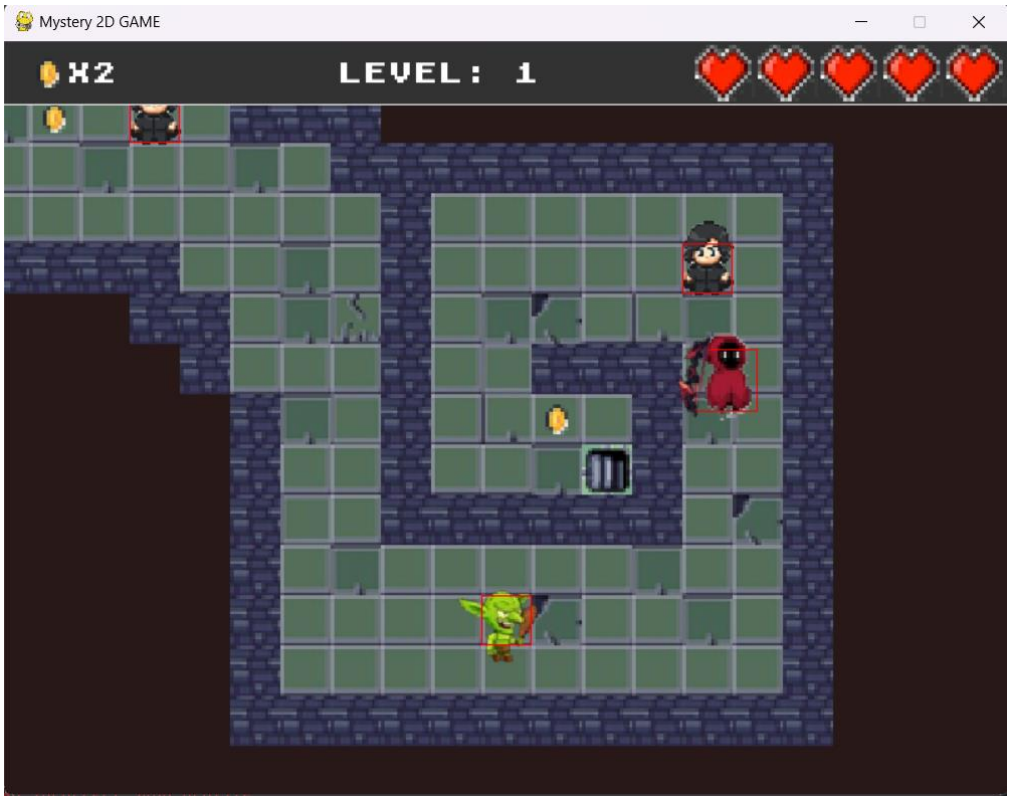
+ Thêm trên map những đồng xu khi nhặt sẽ tính điểm

Tuần 3: Tương tác với AI của từng đối tương

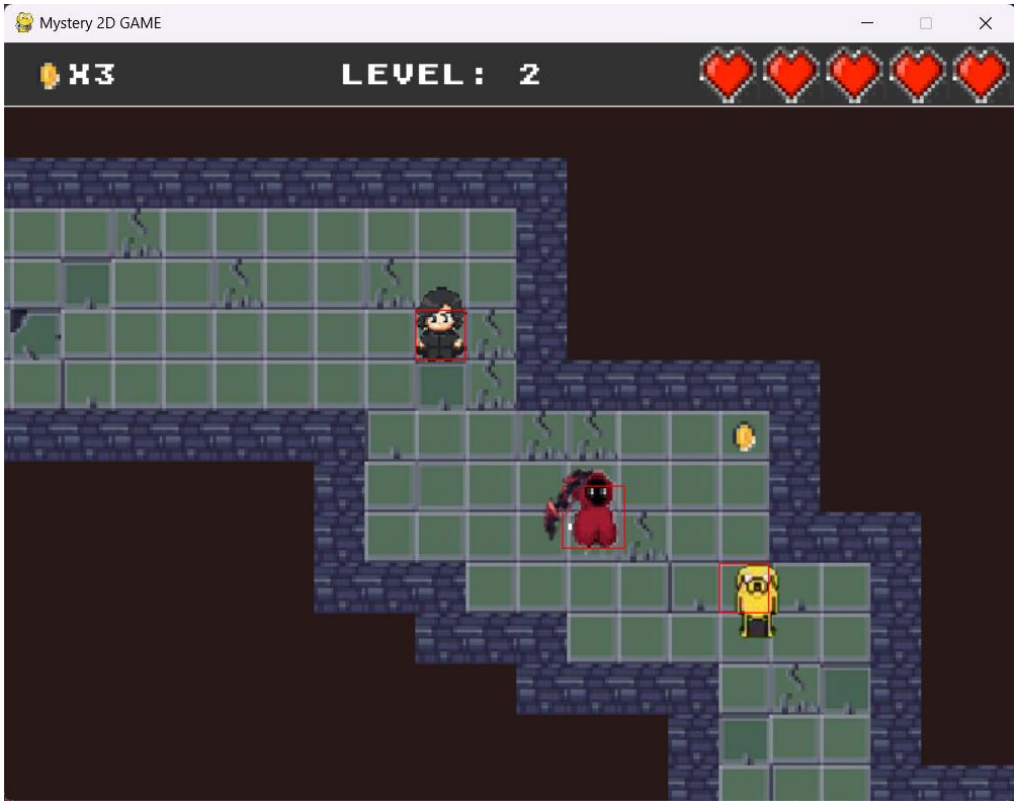
+ Tạo ra 3 màn chơi khác nhau có những đối tượng tương tác với player khác nhau.

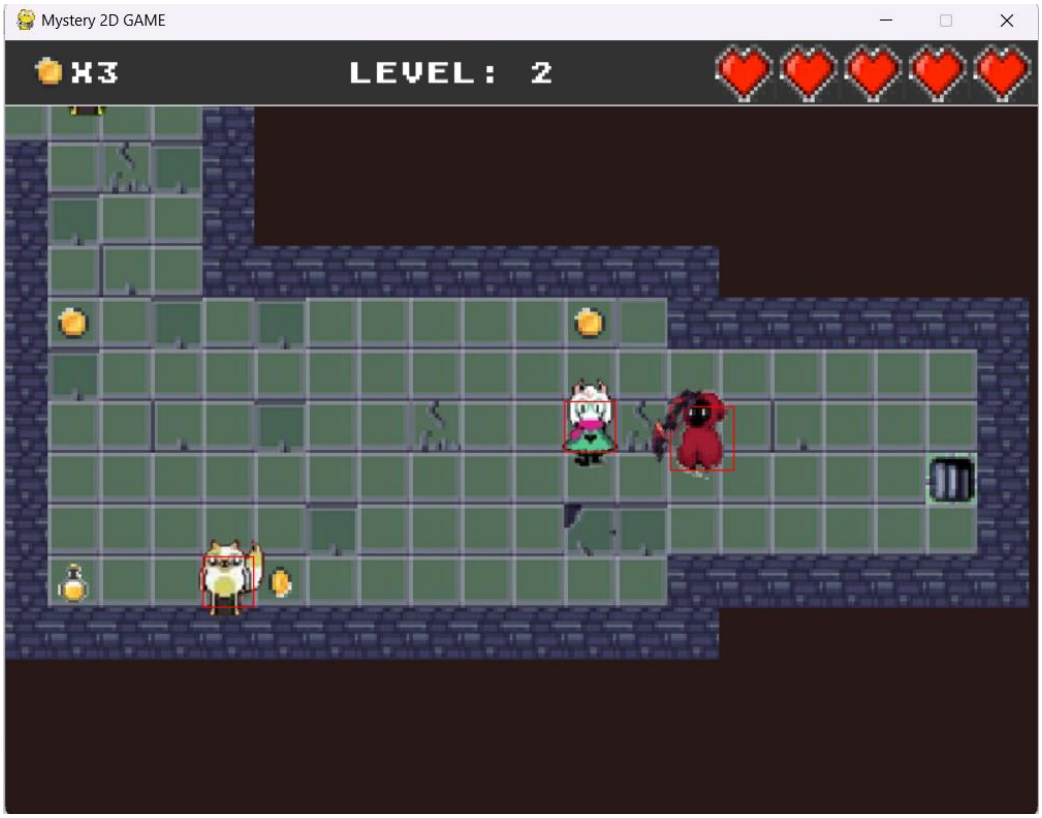
Level 1 :



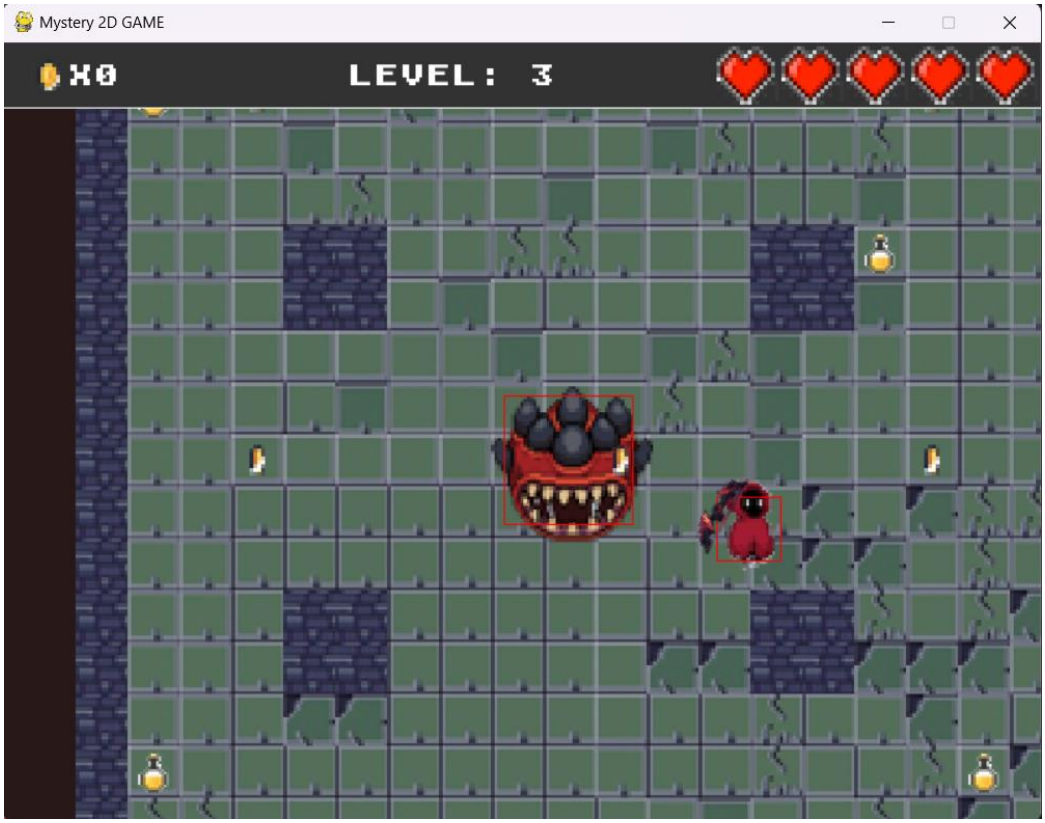


Level 2 :

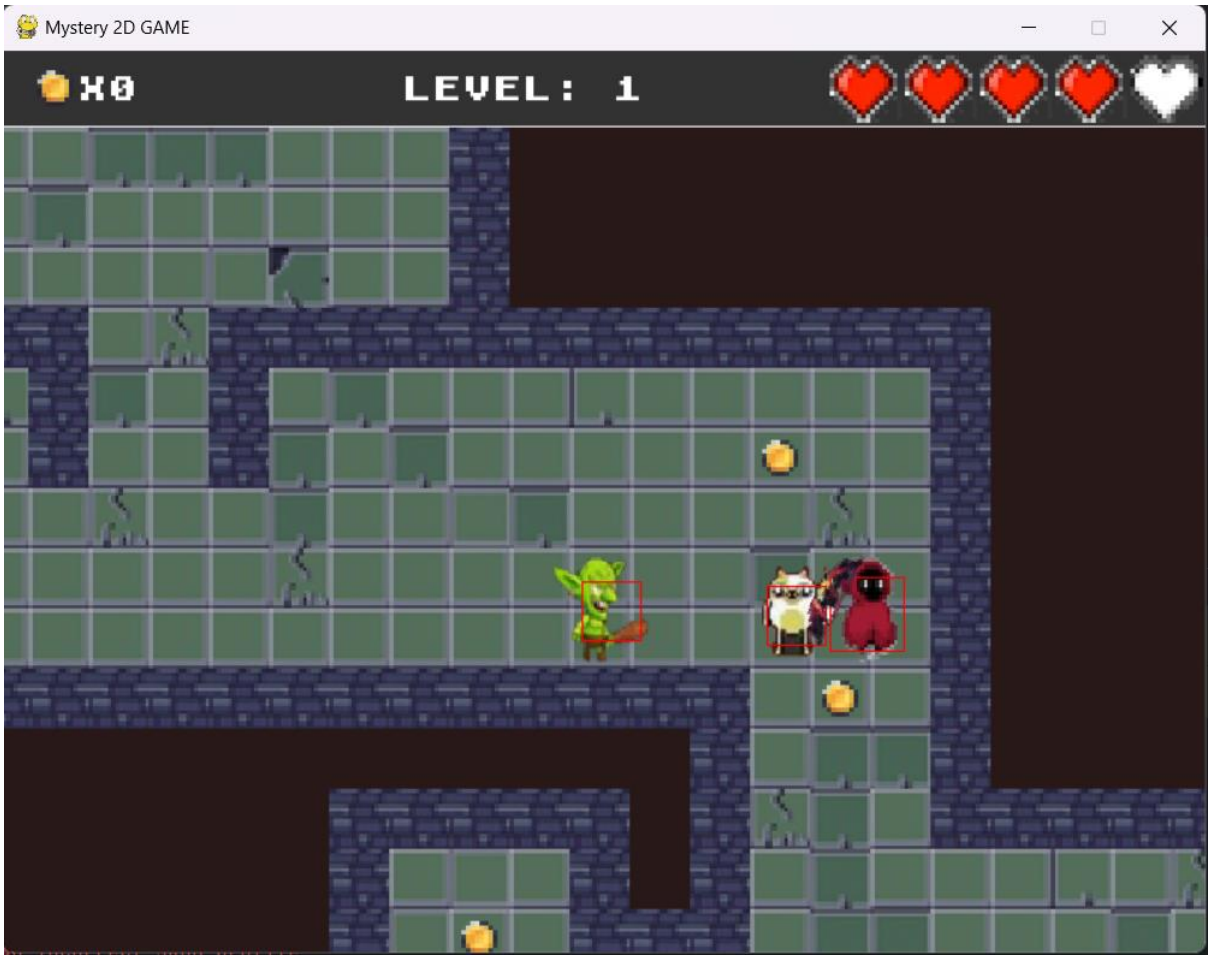




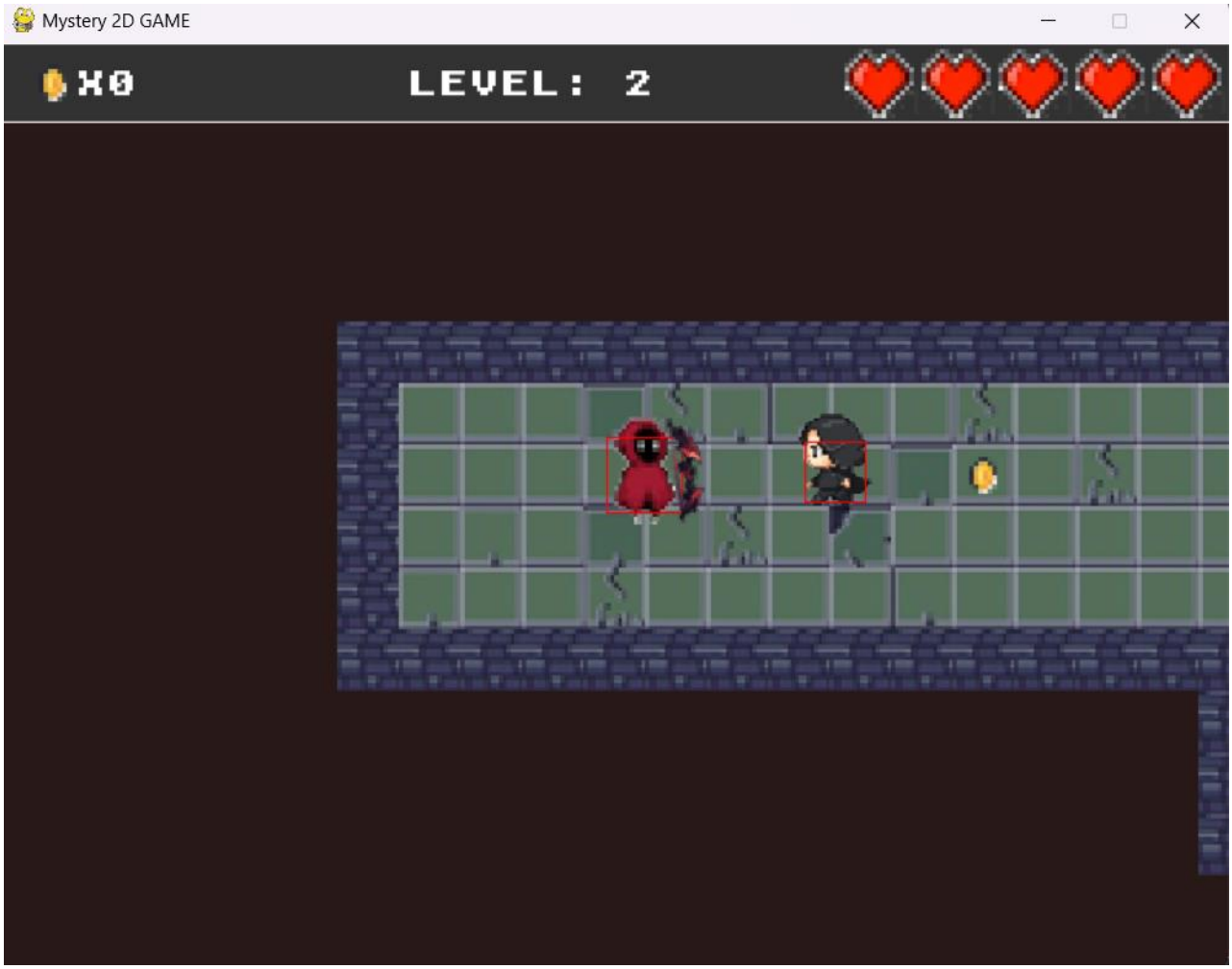
Level 3: Boss Fight



+ Tạo ra AI cho từng đối tượng, sẽ có nhiều hơn 1 đối tượng kẻ thù là goblin , mỗi đối tượng sẽ có thao tác giống nhau ở tốc độ, sát thương (các AI này sẽ đánh tầm gần).



=> Các đối tượng qua level 1 và 2 đánh tay gây sát thương tầm gần



+ Tạo ra BOSS AI đánh tầm xa, sát thương và cách di chuyển khác những con AI bên trên. (Boss phun lửa)

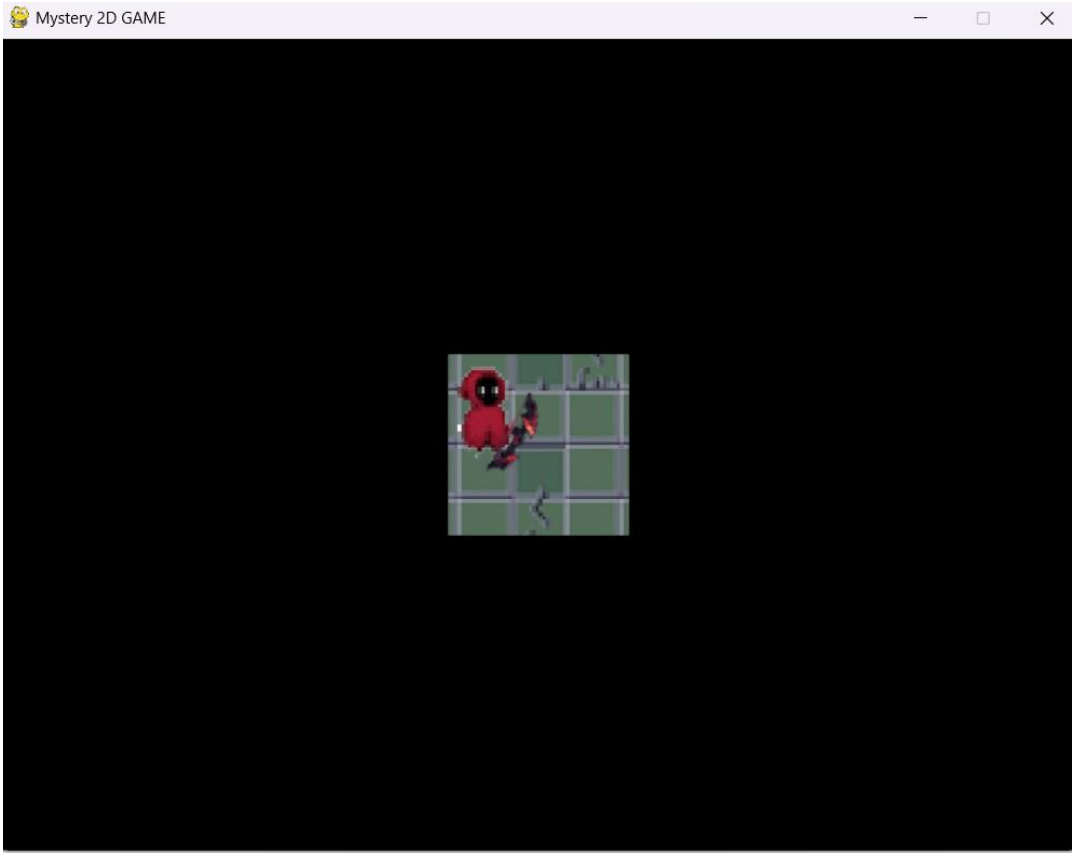
Tuần 4: Hoàn thiện trò chơi

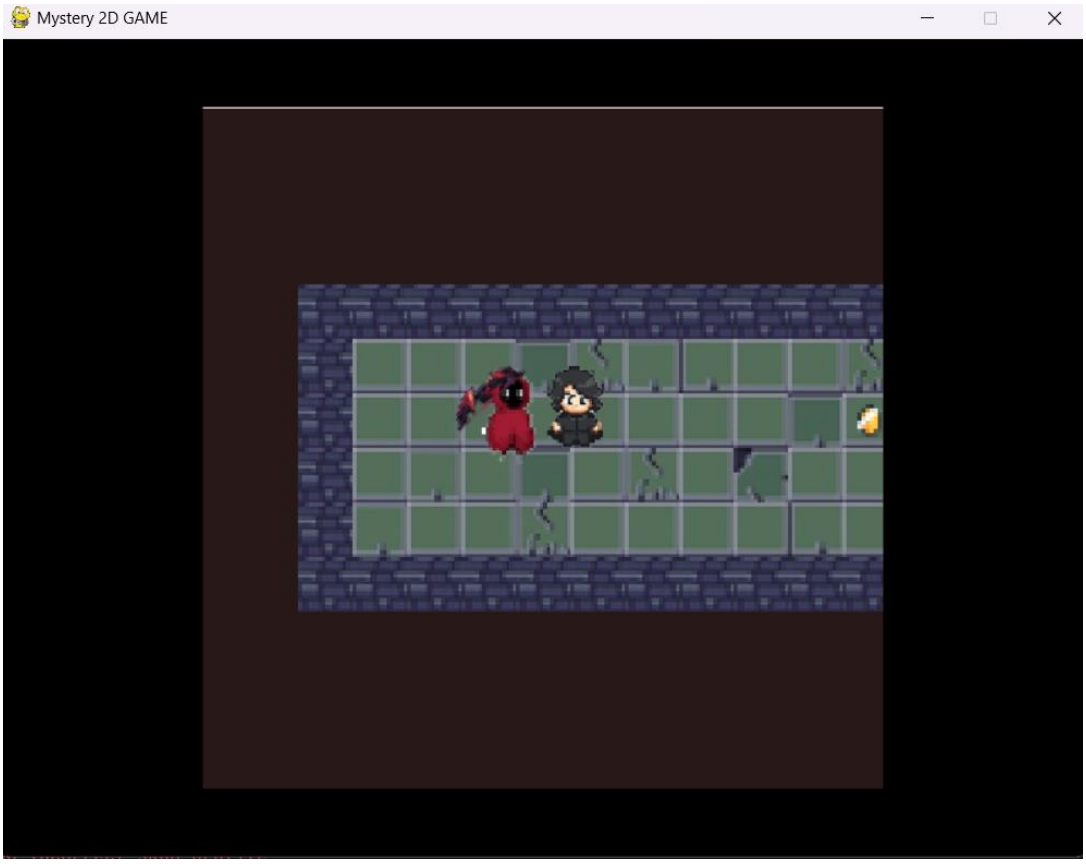
+ Tương tác với các màn chơi , lên các mức level cao hơn bằng cách nhặt xu hoặc đánh quái hoặc di chuyển tới địa điểm cần thiết để qua màn.

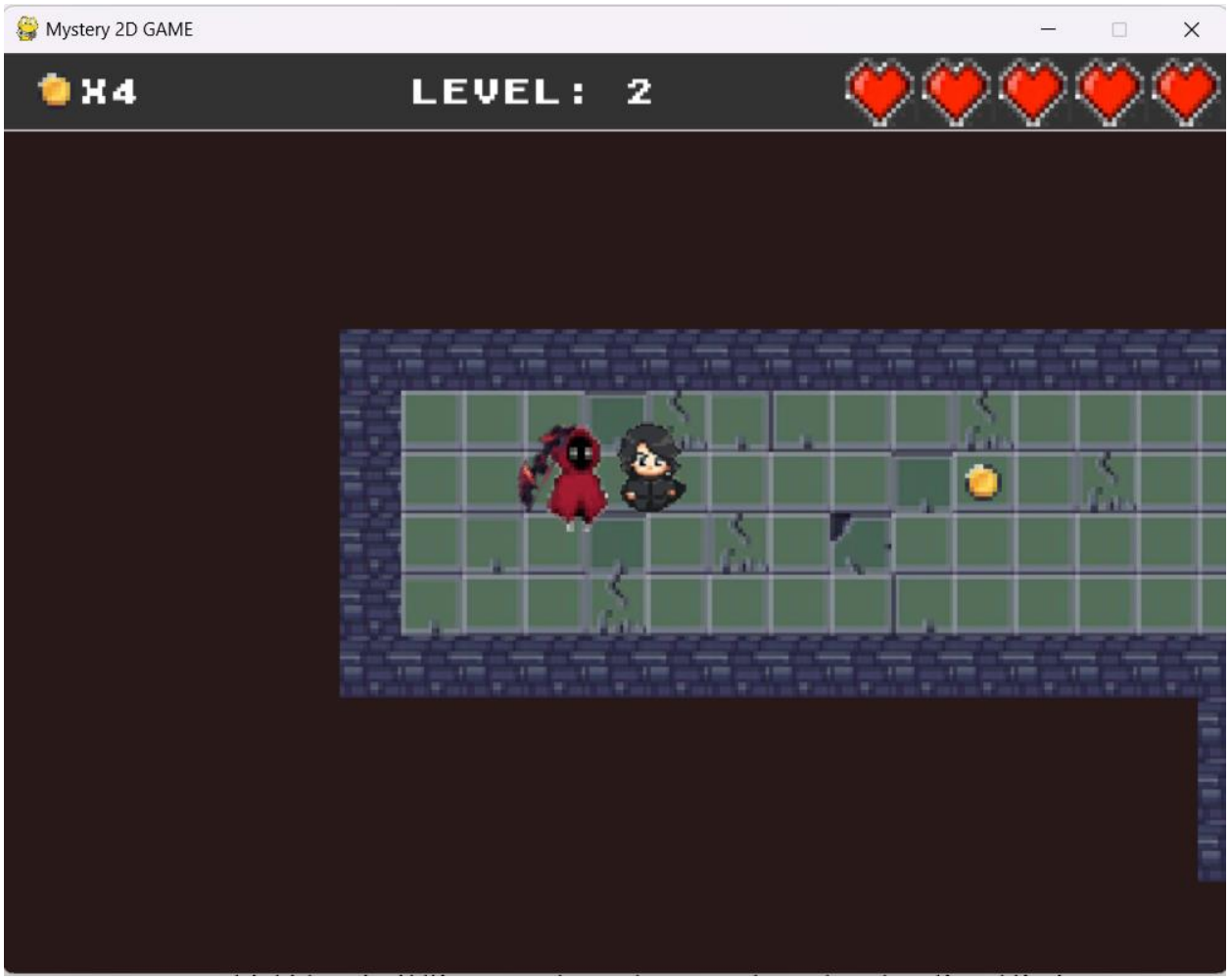
=> Di chuyển tới cái địa điểm cầu thang để qua màn

+ Thêm hiệu ứng mỗi khi qua màn mới bằng hoạt ảnh chuyển cảnh

=> Khi đi tới cầu thang thì sẽ xuất hiện chuyển cảnh qua Map thứ 2 có kèm theo hoạt ảnh chuyển cảnh màn hình đen mở rộng ra theo 4 hướng từ giữa màn hình.

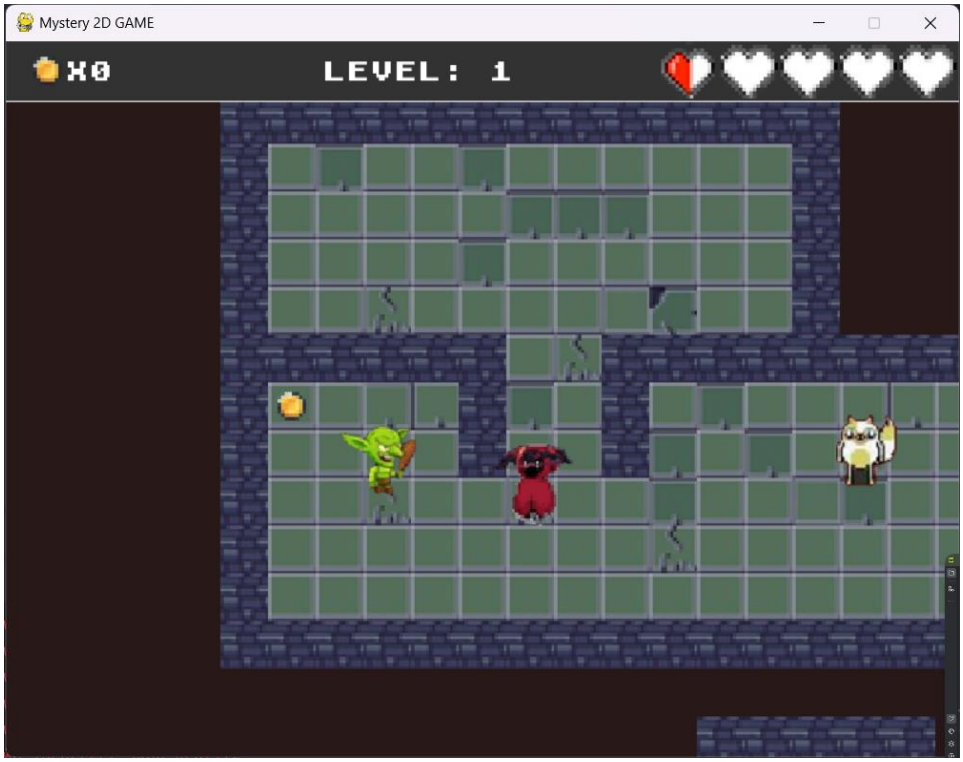

Mystery 2D GAME


Mystery 2D GAME

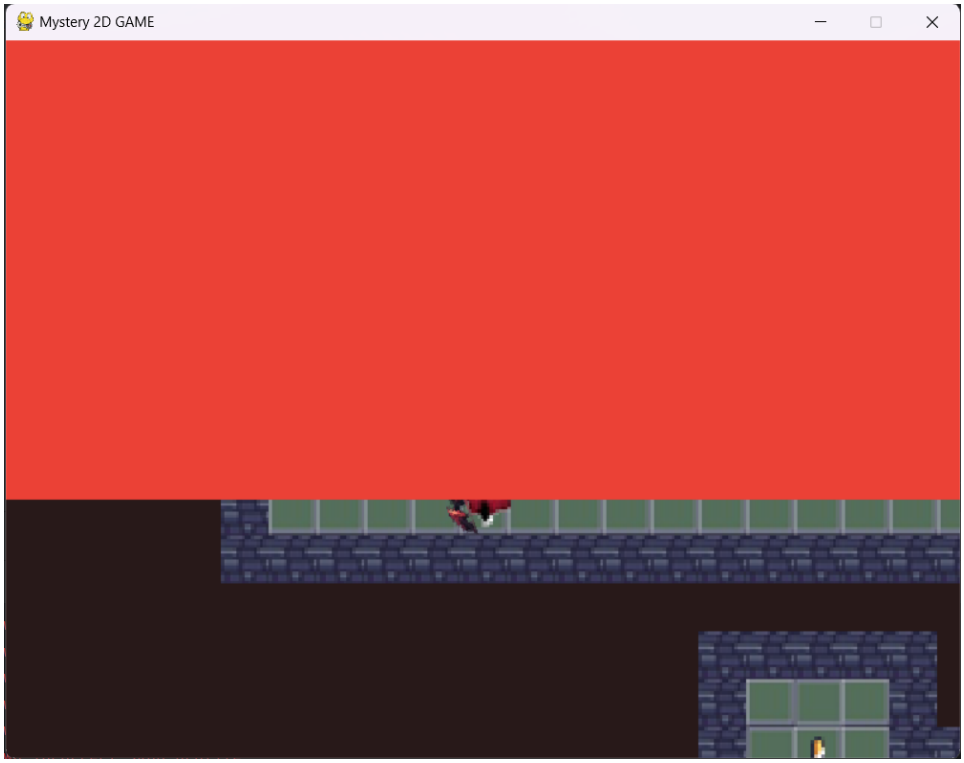


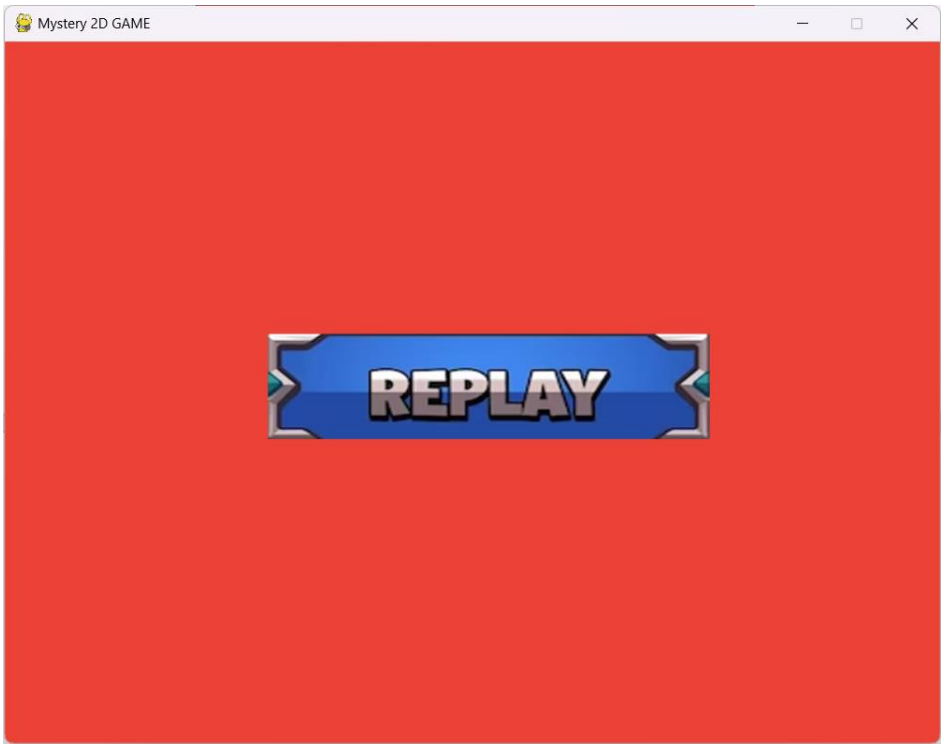
+ Tạo ra nút ấn khi chết sẽ hiện ra chữ restart lại màn chơi

=> Có hiệu ứng màn ảnh đỏ đi xuống và hiển thị nút ấn REPLAY

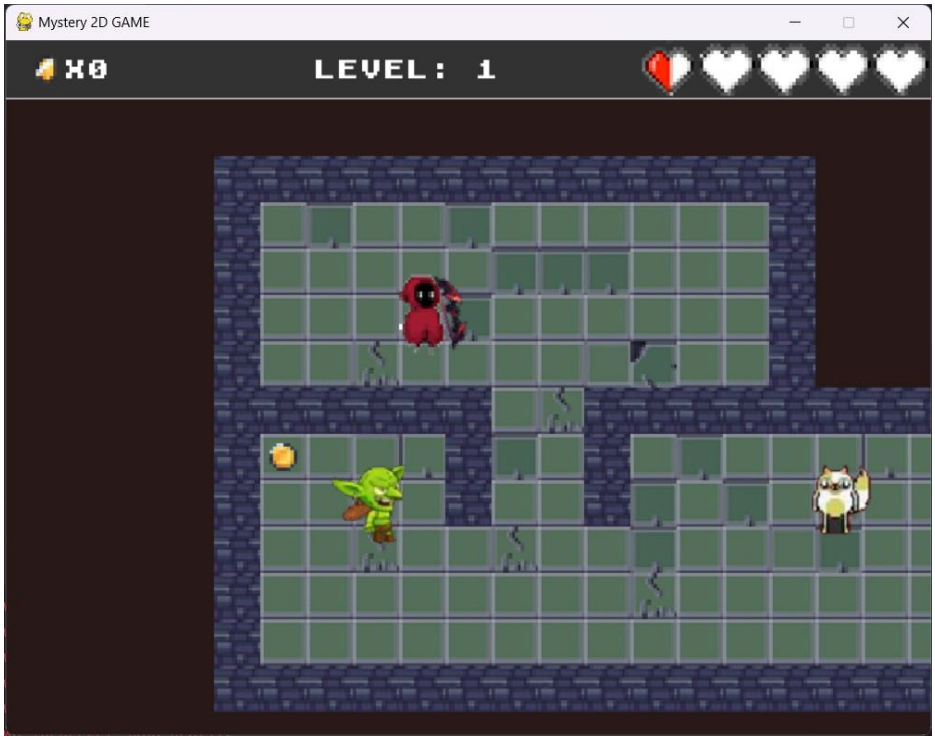


=> Bị đối tượng đánh chết và hiển thị hoạt ảnh màn hình đỏ và nút reset



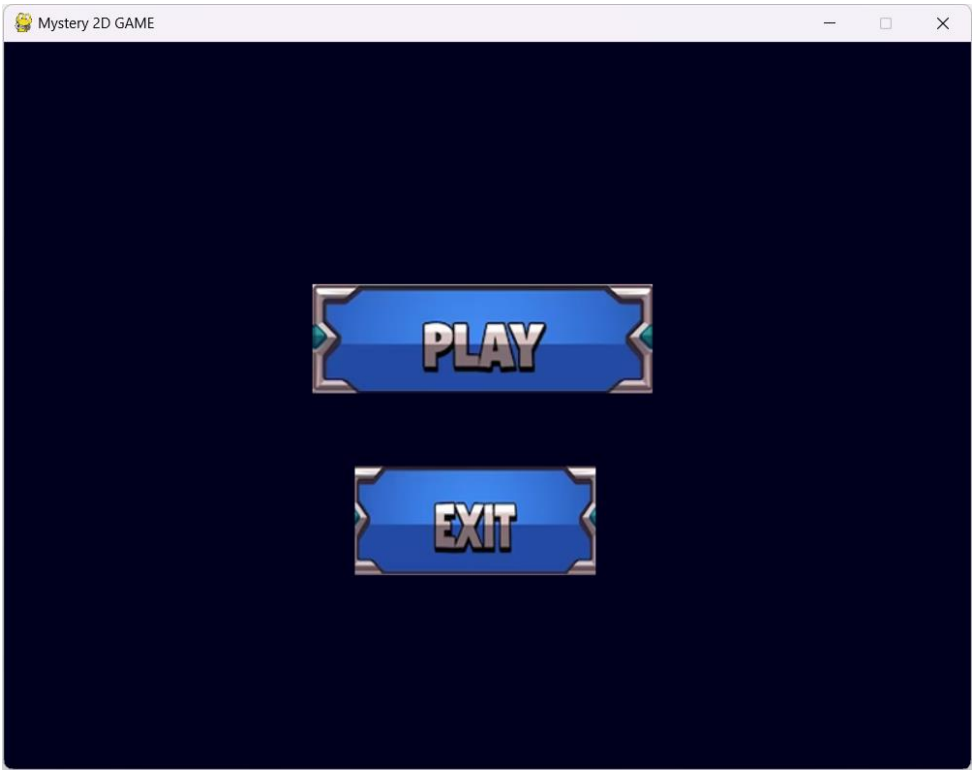


=> Khi ấn nút REPLAY sẽ quay lại màn chơi đó (tại level đã chết)

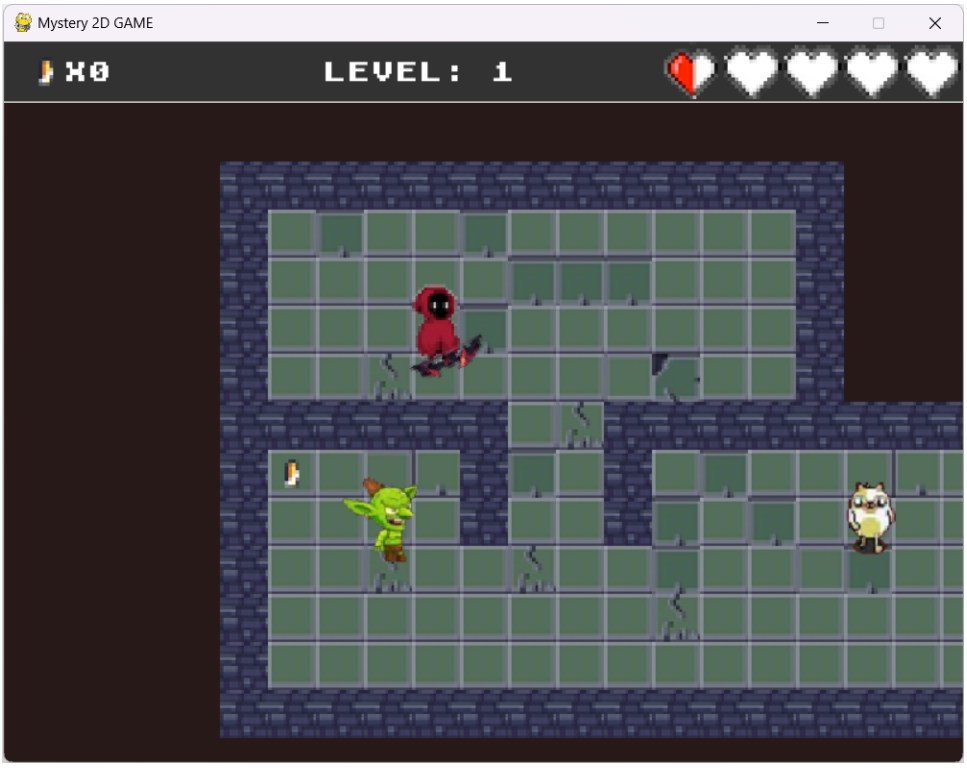


+ Tạo ra Game menu

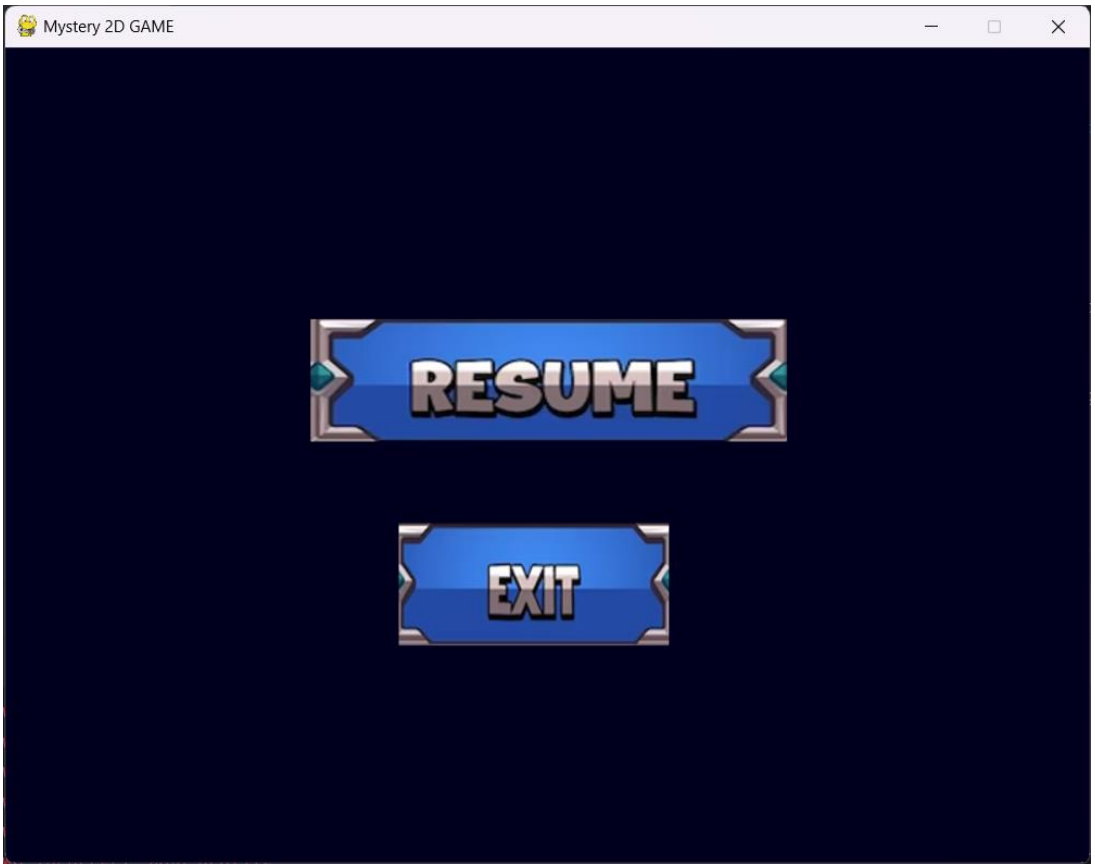
=> Nhấn Start để bắt đầu trò chơi và Exit để thoát trò chơi khi mở game lên,



=> Ấn play để mở game



=> Ấn Escape để pause màn hình , nếu muốn tiếp tục chơi thì ấn RESUME nếu không ta ấn EXIT



+ Thêm Sound effect vào trò chơi, ví dụ xuyên suốt trò chơi sẽ có nhạc, khi đánh kẻ thù hoặc nhặt vật phẩm sẽ có sound nhạc effect bật lên

=> Xem demo bên dưới để có thể xem video có music và sound effect

Game_week4 - Google Drive

4. Tài liệu tham khảo

Chú thích : Các tài liệu ASSET image được cắt ra bằng photoshop để tạo hiệu ứng nhân vật

ASSET image : (những nhân vật, vật phẩm sẽ xuất hiện qua từng màn)

Nhân vật:

Pixelart Idle and Run animations, angrysnail | Pixel art characters, Pixel art

games, Pixel art tutorial (pinterest.com)

Cung :

Pixelart Idle and Run animations, angrysnail | Pixel art characters, Pixel art games, Pixel art tutorial (pinterest.com)

Mũi tên:

WONDERS EX _ game / 2D Game concept art, 2009 Pixel Art on Behance | Undertale pixel art, Cool pixel art, Game concept art (pinterest.com)

Goblin:

WONDERS EX _ game / 2D Game concept art, 2009 Pixel Art on Behance | Undertale pixel art, Cool pixel art, Game concept art (pinterest.com)

Monster:

Hooded Protagonist Animated Character by Penzilla | Pixel art games, Pixel art design, Pixel art characters (pinterest.com)

Steal:

Pin de Elly Herrin en Art en 2024 | Arte de personajes, Arte, Caricaturas (pinterest.com)

Human_dog :

Pin by Laura Watts on Adventure Time | Pixel art characters, Pixel art design, Pixel art tutorial (pinterest.com)

Human_cat:

Pin by Laura Watts on Adventure Time | Pixel art characters, Pixel art design, Pixel art tutorial (pinterest.com)

Dirt :

Pin by Laura Watts on Adventure Time | Pixel art characters, Pixel art design, Pixel art tutorial (pinterest.com)

Boss:

Lava Wurm, James Maxwell | Pixel art games, Pixel art design, Pixel art tutorial (pinterest.com)

Coin:

Pixel art coin vector image on VectorStock | Pixel art design, Pixel art characters, Pixel art tutorial (pinterest.com)

Heart:

Pixel art coin vector image on VectorStock | Pixel art design, Pixel art characters, Pixel art tutorial (pinterest.com)

Potion_red:

Pixel art coin vector image on VectorStock | Pixel art design, Pixel art characters, Pixel art tutorial (pinterest.com)

MAP:

Dungeon Tileset | Indie game art, Pixel art, Game design (pinterest.com)

Fireball:

AoE Templates for Common 5e Fire Spells from 2-Minute Tabletop (2minutetabletop.com)

Button:

Premium Vector | Game ui set of buttons. gui design to build 2d games. vector illustration (freepik.com)

MUSIC:

Arrow_hit : Arrow Hit Sound Effect (youtube.com)

Arrow_shot: Something being hit - Sound Effect (youtube.com)

Coin: Retro Game Coin Sound Effect (youtube.com)

Heal: Heal Sound Effect (youtube.com)

Music game: Toothless Dancing Meme [NEW VARIATIONS] (youtube.com)

Link : Các Yếu Tố Lập Trình Game Cơ Bản Với Pygame (codelearn.io)

Creating a Zelda style game in Python [with some Dark Souls elements] (youtube.com)

Hướng dẫn tạo trò chơi đơn giản với Pygame (niithanoi.edu.vn)

5. Phụ lục 1: Giới thiệu (demo) kết quả (Có trong link GITHUB, các video được để trong thư mục Video_Bao_Cao)

   LINK GITHUB

   DNoman/2D-GAME (github.com)

   Video kết quả Tuần 1:

   Game_week1 - Google Drive

   Video kết quả Tuần 2:

   Game_week2 - Google Drive

   Video kết quả Tuần 3:

   Game_week3 - Google Drive

   Video kết quả Tuần 4:

   Game_week4 - Google Drive

6. Phụ lục 2: docstring (docstring được chú thích ở các module, sử dụng module docstring.py để đọc chú thích docstring)

   LINK GITHUB

   DNoman/2D-GAME (github.com)

   Module main.py :

   + Giải thích : hàm main.py dùng để khai báo pygame, tạo window pygame,scale image cho phù hợp window game, tạo ra nhiều tác nhân là nhân vật điều khiển và tác nhân cần được tiêu diệt bằng cách chạy vòng lặp liên tục cập nhật animate,máu tương tác giữa các nhân vật theo từng frame.Thêm các hoạt ảnh cho 2 item là bình thuốc và coin. Load map vào trò chơi bằng file csv (tạo map bằng file csv). Map tự tạo đơn giản với dự kiến 3 màn chơi, các nhân tố trong map được xử lý để tương tác lẫn nhau .

   + Docstring module mainpy:

   Mystery 2D Game

Kịch bản này triển khai một trò chơi 2D bằng thư viện Pygame. Nó bao gồm di chuyển nhân vật, sử dụng vũ khí, thu thập vật phẩm, tương tác với kẻ thù, tiến triển qua các cấp độ và các phần giao diện người dùng đồ họa.

Modules:

- pygame: Thư viện chính cho phát triển trò chơi.
- csv: Module để đọc các tệp CSV.
- constants: Module hằng số tùy chỉnh.
- character: Module xác định các nhân vật trong trò chơi.
- weapon: Module xác định vũ khí và mũi tên.
- items: Module xác định các vật phẩm có thể thu thập (2 vật phẩm).
- world: Module xác định thế giới trò chơi.
- button: Module xác định các nút tương tác trên màn hình (START,RESUME,…).

Classes:

- DamageText: Lớp sprite để hiển thị số sát thương.
- Screenfade: Lớp để tạo hiệu ứng chuyển cảnh màn hình.

Functions:

- scale_img: Hàm để tỉ lệ hình ảnh.
- draw_text: Hàm để vẽ văn bản trên màn hình.
- reset_level: Hàm để thiết lập lại cấp độ trò chơi.
- draw_info: Hàm để hiển thị thông tin trò chơi trên màn hình.

Variables:

- screen: Bề mặt hiển thị Pygame để vẽ đồ họa.
- clock: Đối tượng Đồng hồ Pygame để điều khiển tốc độ khung hình.
- level: Số nguyên đại diện cho cấp độ trò chơi hiện tại.
- start_game: Boolean cho biết trò chơi đã bắt đầu chưa.

- pause_game: Boolean cho biết trò chơi đã tạm dừng chưa.
- start_intro: Boolean cho biết màn hình giới thiệu đã xuất hiện chưa.
- screen_scroll: Danh sách chứa giá trị cuộn màn hình theo chiều ngang và dọc.
- moving_left, moving_right, moving_up, moving_down: Boolean cho biết hướng di chuyển của người chơi.
- font: Đối tượng font Pygame để vẽ văn bản.
- shot_fx, hit_fx, coin_fx, heal_fx: Đối tượng âm thanh Pygame cho các hiệu ứng âm thanh khác nhau của trò chơi.(bắn, hit, nhặt coin, nhặt poision)
- start_img, exit_img, restart_img, resume_img: hình ảnh các nút MENU tương tác.
- heart_empty, heart_half, heart_full: Hình ảnh các trái tim đại diện cho máu của người chơi.
- coin_images, red_potion: Hình ảnh đại diện cho các vật phẩm có thể thu thập.
- bow_image, arrow_image: Hình ảnh vũ khí và mũi tên.
- tile_list: Danh sách các đối tượng đại diện cho các loại ô trong thế giới trò chơi.
- world_data: Danh sách đại diện cho bố cục của thế giới trò chơi.
- player: Thể hiện của lớp nhân vật đại diện cho người chơi.
- enemy_list: Danh sách thể hiện của lớp nhân vật đại diện cho kẻ thù.
- mob_animations: Danh sách các danh sách chứa các khung hình hoạt ảnh nhân vật.
- intro_fade, death_fade: Thể hiện của lớp Screenfade để tạo hiệu ứng chuyển cảnh màn hình.
- start_button, exit_button, restart_button, resume_button: Thể hiện của lớp Button để tạo các nút tương tác.

Main loop run:

- Chạy mã để bắt đầu trò chơi.
- Nhấn phím START để bắt đầu trò chơi.

- Nhấn phím EXIT để thoát trò chơi.
- Nhấn phím Escape để tạm dừng trò chơi.
- Nhấn phím REPLAY để chơi lại khi nhân vật chết
- Sử dụng các phím mũi tên để di chuyển nhân vật.
- Cập nhật nhân vật, đối tượng và các Item.
- Tương tác với các đối tương hoặc items tạo sound effect
- Qua màn chơi khi di chuyển nhân vật đến vị trí được khởi tạo

+ Code:

```
import pygame
import csv
from pygame import mixer
from constants import *
from character import character
from weapon import Weapon
from items import Item
from world import World
from button import Button

mixer.init()
pygame.init()

screen = pygame.display.set_mode((SCREEN_WIDTH,SCREEN_HEIGHT))
pygame.display.set_caption("Mystery 2D GAME")

#tao ra clock giu cho frame ko di chuyen
clock = pygame.time.Clock()
#level cua game
level = 1
start_game = False
pause_game = False
start_intro = False
screen_scroll = [0,0]
#tao ra di chuyen nhan vat
moving_left = False
moving_right = False
moving_up = False
moving_down = False

#Font chu
font = pygame.font.Font("assets1/fonts/AtariClassic.ttf",20)

###ham scale image
def scale_img(image,scale):
    w = image.get_width()
    h = image.get_height()
    return pygame.transform.scale(image,(w * scale, h * scale))

#load nhac va sound effect
pygame.mixer.music.load("assets1/audio/music_effect.mp3")
```

```
pygame.mixer.music.set_volume(0.3)
pygame.mixer.music.play(-1,0.0,5000)
shot_fx = pygame.mixer.Sound("assets1/audio/arrow_shot_effect.mp3")
shot_fx.set_volume(0.5)
hit_fx = pygame.mixer.Sound("assets1/audio/arrow_hit_effect.mp3")
hit_fx.set_volume(0.3)
coin_fx = pygame.mixer.Sound("assets1/audio/coin_effect.mp3")
coin_fx.set_volume(0.5)
heal_fx = pygame.mixer.Sound("assets1/audio/heal_effect.mp3")
heal_fx.set_volume(5)

#load hinh anh button
start_img =
scale_img(pygame.image.load("assets1/images/buttons/button_start.png"),BUTTON
_SCALE)
exit_img =
scale_img(pygame.image.load("assets1/images/buttons/button_exit.png"),BUTTON_
SCALE)
restart_img =
scale_img(pygame.image.load("assets1/images/buttons/button_restart.png"),BUTT
ON_SCALE)
resume_img =
scale_img(pygame.image.load("assets1/images/buttons/button_resume.png"),BUTTO
N_SCALE)

#anh trai tim ( lam thanh mau )
heart_empty =
scale_img(pygame.image.load("assets1/images/items/heart_empty.png"),ITEM_SCAL
E)
heart_half =
scale_img(pygame.image.load("assets1/images/items/heart_half.png"),ITEM_SCALE
)
heart_full =
scale_img(pygame.image.load("assets1/images/items/heart_full.png"),ITEM_SCALE
)

#load hinh anh coin
coin_images = []
for x in range(4):
    img =
scale_img(pygame.image.load(f"assets1/images/items/coin_f{x}.png"),ITEM_SCALE
)
    coin_images.append(img)

#load hinh anh potion
red_potion =
scale_img(pygame.image.load("assets1/images/items/potion_red.png"),POTION_SCA
LE)
item_images = []
item_images.append(coin_images)
item_images.append(red_potion)


#load hinh anh vu khi
bow_image =
scale_img(pygame.image.load("assets1/images/weapons/bow.png"),WEAPON_SCALE )
arrow_image =
```

```
scale_img(pygame.image.load("assets1/images/weapons/arrow.png"),WEAPON_SCALE
)
fireball_image =
scale_img(pygame.image.load("assets1/images/weapons/fireball.png"),WEAPON_SCA
LE )

#load pixel map
tile_list = []
for x in range(TILE_TYPES):
    tile_image =
pygame.image.load(f"assets1/images/tiles/{x}.png").convert_alpha()
    tile_image = pygame.transform.scale(tile_image, (TILE_SIZE,TILE_SIZE))
    tile_list.append(tile_image)

#ham hien thi diem thuong
def draw_text(text,font,text_col,x,y):
    img = font.render(text,True,text_col)
    screen.blit(img, (x,y))


# def draw_grid():
#     for x in range(30):
#         pygame.draw.line(screen, WHITE, (x *
TILE_SIZE,0),(x*TILE_SIZE,SCREEN_HEIGHT))
#         pygame.draw.line(screen, WHITE, (0,x * TILE_SIZE), (SCREEN_WIDTH,x
* TILE_SIZE))

#hien thi so mau mat di
class DamageText(pygame.sprite.Sprite):
    def __init__(self,x,y,damage,color):
        pygame.sprite.Sprite.__init__(self)
        self.image = font.render(damage,True,color)
        self.rect = self.image.get_rect()
        self.rect.center = (x,y)
        self.counter = 0
    def update(self):
        # tao lai vi tri khi screen scroll
        self.rect.x += screen_scroll[0]
        self.rect.y += screen_scroll[1]
        #xoa text sau moi lan update
        self.rect.y -= 1
        #xoa counter sau vai giay
        self.counter +=1
        if self.counter > 30:
            self.kill()
#Class hoat anh chuyen canh moi level
class Screenfade():
    def __init__(self,direction,colour,speed):
        self.direction = direction
        self.colour =colour
        self.speed = speed
        self.fade_counter = 0
    def fade(self):
        fade_complete = False
        self.fade_counter += self.speed
        if self.direction == 1:
            pygame.draw.rect(screen,self.colour,(0 -
```

```
self.fade_counter,0,SCREEN_WIDTH // 2 , SCREEN_HEIGHT))
            pygame.draw.rect(screen,self.colour,(SCREEN_WIDTH // 2 +
self.fade_counter,0,SCREEN_WIDTH,SCREEN_HEIGHT))
            pygame.draw.rect(screen,self.colour,(0,0 -
self.fade_counter,SCREEN_WIDTH,SCREEN_HEIGHT // 2))
            pygame.draw.rect(screen,self.colour,(0, SCREEN_HEIGHT//2 +
self.fade_counter,SCREEN_WIDTH,SCREEN_HEIGHT))

        elif self.direction == 2:
            pygame.draw.rect(screen,self.colour,(0,0,SCREEN_WIDTH,0 +
self.fade_counter))

        if self.fade_counter >= SCREEN_WIDTH:
            fade_complete = True

        return fade_complete

#create empty tile list
# world_data = [
#     [7,7,7,7,7,7],
#     [7,0,1,2,4,7],
#     [7,3,1,2,1,7],
#     [7,6,6,6,2,7],
#     [7,0,0,0,3,7],
#     [7,0,0,0,3,7],
#     [7,0,0,0,3,7]
# ]

world_data = []
for row in range(ROW):
    r = [-1] * COLS
    world_data.append(r)
#tao World
with open(f"levels/level{level}_data.csv",newline="") as csvfile:
    reader = csv.reader(csvfile, delimiter=",")
    for x,row in enumerate(reader):
        for y,tile in enumerate(row):
            world_data[x][y] = int(tile)

#tao vu khi cho nhan vat
bow = Weapon(bow_image,arrow_image)
#tao Sprite group
damage_text_group = pygame.sprite.Group()
arrow_group = pygame.sprite.Group()
item_group = pygame.sprite.Group()
fireball_group = pygame.sprite.Group()

score_coin = Item(SCREEN_WIDTH - 765,25,0,coin_images,True)
item_group.add(score_coin)


# potion = Item(200,200,1,[red_potion])
# item_group.add(potion)
# coin = Item(400,400,0,coin_images)
# item_group.add(coin)

# print(item_group)
```

```
#tao font chu hien thi mau
# damage_text = DamageText(300,400,"15",RED)
# damage_text_group.add(damage_text)

#load hinh anh cac doi tuong trong game
mob_animations = []
mob_types =
["girl","goblin","monster","steal","human_dog","human_cat","Boss"]

animation_type =["idle","run"]
for mob in mob_types:
#load hinh anh
    animation_list =[]
    for animation in animation_type:
        #reset danh sach tam thoi chua hinh anh
        temp_list = []
        for i in range(4):
            img =
pygame.image.load(f"assets1/images/characters/{mob}/{animation}/{i}.png").con
vert_alpha()
            if mob == "goblin" :
                img = scale_img(img,SCAlE_GOBLIN)
            elif mob == "monster":
                img = scale_img(img,SCALE_MONSTER)
            elif mob == "steal":
                img = scale_img(img,SCALE_STEAL)
            elif mob == "human_dog":
                img = scale_img(img,SCALE_STEAL)
            elif mob == "human_cat":
                img = scale_img(img,SCALE_STEAL)
            else :
                img = scale_img(img,SCALE)
            temp_list.append(img)
        animation_list.append(temp_list)
    mob_animations.append(animation_list)

#Ham reset level
def reset_level():
    damage_text_group.empty()
    arrow_group.empty()
    item_group.empty()
    fireball_group.empty()
    data = []
    for row in range(ROW):
        r = [-1] * COLS
        data.append(r)

    return data

#ham hien thi thong tin game
def draw_info():
    pygame.draw.rect(screen,PANEL,(0,0,SCREEN_WIDTH,50))
    pygame.draw.line(screen,WHITE,(0,50),((SCREEN_WIDTH,50)))
    #mang song
    half_heart_drawn = False
    for i in range(5):
```

```
        if player.health >= ((i+1)*20):
            screen.blit(heart_full, (550 + i * 50,0))
        elif (player.health % 20 > 0) and half_heart_drawn == False:
            screen.blit(heart_half, (550 + i * 50, 0))
            half_heart_drawn = True
        else:
            screen.blit(heart_empty, (550 + i * 50, 0))

    #level
    draw_text("LEVEL: "+ str(level),font,WHITE,SCREEN_WIDTH/3,15)
    #diem thuong
    draw_text(f"X{player.score}",font,WHITE,SCREEN_WIDTH-750,15)

#tao the gioi
world = World()
world.process_data(world_data,tile_list,item_images,mob_animations)

#them coin tu level data
for item in world.item_list:
    item_group.add(item)

#Tao nhan vat
player = world.player

#tao ke thu
enemy_list = world.character_list

#Tao screen fade
intro_fade = Screenfade(1,BLACK,4)
death_fade = Screenfade(2,PINK,4)

#Tao Button
start_button = Button(SCREEN_WIDTH//2 - 145,SCREEN_HEIGHT// 2 -
100,start_img)
exit_button = Button(SCREEN_WIDTH//2 - 110,SCREEN_HEIGHT// 2 + 50,exit_img)
restart_button = Button(SCREEN_WIDTH//2 - 175,SCREEN_HEIGHT// 2 -
50,restart_img)
resume_button = Button(SCREEN_WIDTH//2 - 175,SCREEN_HEIGHT// 2 -
100,resume_img)

# main game loop
run = True
while run:

    # dieu khien frame rate
    clock.tick(FPS)
    if start_game == False:
        screen.fill(MENU_BG)
        if start_button.draw(screen):
            start_game = True
        if exit_button.draw(screen):
            run = False
    else:
        if pause_game == True:
            screen.fill(MENU_BG)
            if resume_button.draw(screen):
                pause_game = False
```

```
            if exit_button.draw(screen):
                run = False
        else:
            screen.fill(BG)
            # draw_grid()
            if player.alive:
                #tinh toan buoc di chuyen nhan vat
                dx = 0
                dy = 0
                if moving_right == True:
                    dx = SPEED
                if moving_left == True:
                    dx = -SPEED
                if moving_up == True:
                    dy = -SPEED
                if moving_down == True:
                    dy = SPEED

                #Di chuyen nhan vat
                screen_scroll,level_complete =
player.move(dx,dy,world.obstacle_tiles,world.exit_tile)


                #Cap nhat tat ca doi tuong
                world.update(screen_scroll)
                for enemy in enemy_list:
                    fireball = enemy.ai(player ,
world.obstacle_tiles,screen_scroll,fireball_image)
                    if fireball:
                        fireball_group.add(fireball)
                    if enemy.alive :
                        enemy.update()
                player.update()
                arrow = bow.update(player)
                bow.update(player)
                if arrow:
                    arrow_group.add(arrow)
                    shot_fx.play()
                for arrow in arrow_group:
                    damage, damage_pos =
arrow.update(screen_scroll,world.obstacle_tiles,enemy_list)
                    if damage:
                        damage_text = DamageText(damage_pos.centerx,
damage_pos.y, str(damage), RED)
                        damage_text_group.add(damage_text)
                        hit_fx.play()
                damage_text_group.update()
                fireball_group.update(screen_scroll,player)
                item_group.update(screen_scroll,player ,coin_fx,heal_fx)

            # print(enemy.health)

            #Tao nhan vat tren man hinh
            world.draw(screen)
            for enemy in enemy_list:
                enemy.draw(screen)
            player.draw(screen)
```

```
            bow.draw(screen)
            for arrow in arrow_group:
                arrow.draw(screen)
            for fireball in fireball_group:
                fireball.draw(screen)
            damage_text_group.draw(screen)
            item_group.draw(screen)
            draw_info()
            score_coin.draw(screen)

            #kiem tra level complete
            if level_complete == True:
                start_intro = True
                level += 1
                world_data = reset_level()
                with open(f"levels/level{level}_data.csv", newline="") as
csvfile:
                    reader = csv.reader(csvfile, delimiter=",")
                    for x, row in enumerate(reader):
                        for y, tile in enumerate(row):
                            world_data[x][y] = int(tile)
                world = World()
                world.process_data(world_data, tile_list, item_images,
mob_animations)
                temp_hp = player.health
                temp_score = player.score
                player = world.player
                player.health = temp_hp
                player.score = temp_score
                enemy_list = world.character_list
                score_coin = Item(SCREEN_WIDTH - 765,25,0,coin_images,True)
                item_group.add(score_coin)
                for item in world.item_list:
                    item_group.add(item)

            #intro
            if start_intro == True:
                if intro_fade.fade():
                    start_intro = False
                    intro_fade.fade_counter = 0

            #death screen
            if player.alive == False:
                if death_fade.fade():
                    if restart_button.draw(screen):
                        death_fade.fade_counter = 0
                        start_intro = True
                        world_data = reset_level()
                        with open(f"levels/level{level}_data.csv",
newline="") as csvfile:
                            reader = csv.reader(csvfile, delimiter=",")
                            for x, row in enumerate(reader):
                                for y, tile in enumerate(row):
                                    world_data[x][y] = int(tile)
                        world = World()
                        world.process_data(world_data, tile_list,
item_images, mob_animations)
```

```
                        player = world.player
                        enemy_list = world.character_list
                        score_coin = Item(SCREEN_WIDTH - 765, 25, 0,
coin_images, True)
                        item_group.add(score_coin)
                        for item in world.item_list:
                            item_group.add(item)

    #duy tri window game
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False

    # an nut di chuyen
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_a:
                moving_left = True
            if event.key == pygame.K_d:
                moving_right = True
            if event.key == pygame.K_w:
                moving_up = True
            if event.key == pygame.K_s:
                moving_down = True
            if event.key == pygame.K_ESCAPE:
                pause_game = True

    # tha nut di chuyen
        if event.type == pygame.KEYUP:
            if event.key == pygame.K_a:
                moving_left = False
            if event.key == pygame.K_d:
                moving_right = False
            if event.key == pygame.K_w:
                moving_up = False
            if event.key == pygame.K_s:
                moving_down = False


    pygame.display.update()
pygame.quit()
```

Module Character.py :

+ Giải thích: Dùng để tạo ra đặc tính của đối tượng (tạo ra máu, action, animate của đối tượng khi đang đứng yên hoặc chạy,cập nhật lại trạng thái của nhân vật qua từng frame, tạo nhân vật lên window pygame qua từng frame). Tạo ra camera scroll để nhân vật di chuyển tự do khắp bản đồ.

+ Docstring module Character :

Class Character:

Đại diện cho một nhân vật trong trò chơi.

Attributes:

- x (int): Tọa độ x của nhân vật trên màn hình.
- y (int): Tọa độ y của nhân vật trên màn hình.
- health (int): Mức điểm máu của nhân vật.
- mob_animations (dict): Danh sách các animation của nhân vật.
- char_type (int): Loại nhân vật.
- boss (bool): Xác định xem nhân vật có phải là boss hay không.
- size (int): Kích thước của nhân vật.
- score (int): Điểm số của nhân vật.
- animation_list (list): Danh sách các frame trong animation của nhân vật.
- flip (bool): Xác định xem nhân vật có đang bị lật ngược hay không.
- frame_index (int): Chỉ số của frame hiện tại đang được sử dụng trong animation.
- action (int): Hành động hiện tại của nhân vật (0:hoạt ảnh đứng yên, 1:hoạt ảnh chạy).
- running (bool): Xác định xem nhân vật có đang chạy hay không.
- alive (bool): Xác định xem đối tượng còn sống hay không.
- hit (bool): Xác định xem nhân vật đã bị tấn công hay không và đối tượng bị tấn công hay không.
- last_hit (int): Thời điểm nhân vật bị tấn công gần nhất.
- last_attack (int): Thời điểm cuối cùng boss tấn công.
- stunned (bool): Xác định xem đối tượng có bị choáng hay không.
- update_time (int): Thời điểm cập nhật gần nhất của animation.
- image (list): Hình ảnh hiện tại của nhân vật.
- rect :hình chữ nhật bao quanh nhân vật trên màn hình.

Methods:

- move(dx, dy, obstacle_tile,exit_tile): Di chuyển nhân vật trên màn hình, di chuyển đến địa điểm chỉ định được qua màn chơi khác

- ai(player, obstacle_tiles, screen_scroll, fireball_image): Xử lý trí tuệ nhân tạo cho nhân vật.

- update(): Cập nhật trạng thái của nhân vật.

- update_action(new_action): Cập nhật hành động của nhân vật.

- draw(surface): Vẽ nhân vật lên màn hình.

+ Code:

```
import math

import pygame
import math
import weapon

import constants
from constants import *
class character():
    def __init__(self,x,y,health,mob_animations,char_type,boss,size):
        self.char_type = char_type
        self.boss = boss
        self.score = 0
        self.animation_list = mob_animations[char_type]
        self.flip = False
        self.frame_index = 0
        self.action = 0 # 0: hoat anh dung yen,1: hoat anh chay
        self.running = False
        self.health = health
        self.alive = True
        self.hit = False
        self.last_hit = pygame.time.get_ticks()
        self.last_attack = pygame.time.get_ticks()
        self.stunned = False

        self.update_time = pygame.time.get_ticks()
        self.image = self.animation_list[self.action][self.frame_index]
        self.rect = pygame.Rect(0,0,TILE_SIZE  *size,TILE_SIZE * size)
        self.rect.center = (x,y)

    def move(self,dx,dy,obstacle_tile,exit_tile = None):
        screen_scroll = [0,0]
        level_complete = False
        self.running = False
        if dx != 0 or dy != 0:
            self.running = True
        if dx < 0:
            self.flip = True
        if dx > 0:
            self.flip = False

        #dieu khien duong cheo
        if dx != 0 and dy !=0:
            dx = dx * (math.sqrt(2)/2)
            dy = dy * (math.sqrt(2) / 2)
```

```
        self.rect.x += dx
        for obstacle in obstacle_tile:
            #kiem tra va cham trai va phai
            if obstacle[1].colliderect(self.rect):
                if dx > 0:
                    self.rect.right = obstacle[1].left
                if dx < 0:
                    self.rect.left = obstacle[1].right
        self.rect.y += dy
        for obstacle in obstacle_tile:
            #kiem tra va cham tren va duoi
            if obstacle[1].colliderect(self.rect):
                if dy > 0:
                    self.rect.bottom = obstacle[1].top
                if dy < 0:
                    self.rect.top = obstacle[1].bottom

        if self.char_type == 0:
            # kiem tra nhan vat tuong tac voi exit gate hay chua
            if exit_tile[1].colliderect(self.rect):
                #dam bao rang player den gan voi loi exit
                exit_dist = math.sqrt(((self.rect.centerx -
exit_tile[1].centerx) ** 2) + ((self.rect.centery - exit_tile[1].centery) **
2))
                if exit_dist < 30:
                    level_complete = True
            #cap nhat scoll (di chuyen camera)
            # trai va phai
            if self.rect.right > (SCREEN_WIDTH - SCROLL_THRESH):
                screen_scroll[0] = (SCREEN_WIDTH - SCROLL_THRESH) -
self.rect.right
                self.rect.right = SCREEN_WIDTH - SCROLL_THRESH
            if self.rect.left < SCROLL_THRESH:
                screen_scroll[0] = SCROLL_THRESH - self.rect.left
                self.rect.left = SCROLL_THRESH

            #len va xuong
            if self.rect.bottom > (SCREEN_HEIGHT - SCROLL_THRESH):
                screen_scroll[1] = (SCREEN_HEIGHT - SCROLL_THRESH) -
self.rect.bottom
                self.rect.bottom = SCREEN_HEIGHT - SCROLL_THRESH
            if self.rect.top < SCROLL_THRESH:
                screen_scroll[1] = SCROLL_THRESH - self.rect.top
                self.rect.top = SCROLL_THRESH

            return screen_scroll,level_complete

    def ai(self,player,obstacle_tiles,screen_scroll,fireball_image):
        clipped_line = []
        stun_cooldown = 100
        ai_dx = 0
        ai_dy = 0
        fireball = None

        #dinh vi lai mob dua tren scroll man hinh
        self.rect.x += screen_scroll[0]
```

```
        self.rect.y += screen_scroll[1]

        #tao ra duong quan sat cua doi tuong
        line_of_sight = ((self.rect.centerx,self.rect.centery),
(player.rect.centerx, player.rect.centery))
        #kiem tra xem duong quan sat qua duoc buc tuong hay khong
        for obstacle in obstacle_tiles:
            if obstacle[1].clipline(line_of_sight):
                clipped_line = obstacle[1].clipline(line_of_sight)

        #toa do cua doi tuong so voi player, khien cac doi tuong huong ve
player
        #kiem tra khoang cach giua doi tuong va player
        dist = math.sqrt(((self.rect.centerx - player.rect.centerx)**2) +
((self.rect.centery - player.rect.centery)**2))
        if not clipped_line and dist > RANGE:
            if self.rect.centerx > player.rect.centerx:
                ai_dx = -ENEMY_SPEED
            if self.rect.centerx < player.rect.centerx:
                ai_dx = ENEMY_SPEED
            if self.rect.centery > player.rect.centery:
                ai_dy = -ENEMY_SPEED
            if self.rect.centery < player.rect.centery:
                ai_dy = ENEMY_SPEED

        if self.alive:
            if not self.stunned:

                #Tien toi player
                self.move(ai_dx, ai_dy,obstacle_tiles)

                #tan cong player
                if dist < ATTACK_RANGE and player.hit == False:
                    player.health -= 10
                    player.hit = True
                    player.last_hit =pygame.time.get_ticks()

                #boss enemies fireball
                fireball_cooldown = 700
                if self.boss:
                    if dist < 500:
                        if pygame.time.get_ticks() - self.last_attack >=
fireball_cooldown:
                            fireball =
weapon.Fireball(fireball_image,self.rect.centerx,self.rect.centery,player.rec
t.centerx,player.rect.centery)
                            self.last_attack = pygame.time.get_ticks()

            #kiem tra xem hit True hay False
            if self.hit ==True:
                self.hit = False
                self.last_hit = pygame.time.get_ticks()
                self.stunned = True
                self.update_action(0)

            if (pygame.time.get_ticks() - self.last_hit > stun_cooldown):
                self.stunned = False
```

```
        return fireball
    def update(self):
        #kiem tra xem doi tuong da chet hay khong
        if self.health <= 0:
            self.health = 0
            self.alive = False
        #reset thoi gian attack player
        hit_cooldown =500
        if self.char_type == 0:
            if self.hit == True and pygame.time.get_ticks() - self.last_hit >
hit_cooldown:
                self.hit = False
        #kiem tra nhan vat dang o trang thai action nao
        if self.running == True:
            self.update_action(1) #1 : hoat anh chay
        else:
            self.update_action(0) #0 : hoat anh dung yen
        animation_cooldown = 70
        # tao ra animation
        #update hinh anh
        self.image = self.animation_list[self.action][self.frame_index]
        if pygame.time.get_ticks() - self.update_time > animation_cooldown:
            self.frame_index +=1
            self.update_time = pygame.time.get_ticks()
        if self.frame_index >= len(self.animation_list[self.action]):
            self.frame_index = 0

    def update_action(self,new_action):
        #kiem tra action moi co khac so voi action cu
        if new_action != self.action:
            self.action = new_action
        # #cap nhat animation
        # self.frame_index = 0
        # self.update_time = pygame.time.get_ticks()

    def draw(self,surface):
        flipped_image = pygame.transform.flip(self.image,self.flip,False)
        if self.char_type == 0:
            surface.blit(flipped_image, (self.rect.x + SCALE*2,self.rect.y -
SCALE * OFFSET))
        elif self.char_type == 1:
            surface.blit(flipped_image,(self.rect.x - SCALE * 8,self.rect.y -
SCALE * 20))
        elif self.char_type == 2:
            surface.blit(flipped_image, (self.rect.x- SCALE * 6,self.rect.y -
SCALE * 7))
        elif self.char_type == 3:
            surface.blit(flipped_image, (self.rect.x- SCALE * 4,self.rect.y -
SCALE * 7))
        elif self.char_type == 4:
            surface.blit(flipped_image, (self.rect.x- SCALE *3,self.rect.y -
SCALE * 5))
        elif self.char_type == 5:
            surface.blit(flipped_image, (self.rect.x- SCALE *3,self.rect.y -
SCALE * 5))
        elif self.char_type == 6:
            surface.blit(flipped_image, (self.rect.x- SCALE * 4,self.rect.y -
```

```
SCALE * 4))
        else :
            surface.blit(flipped_image, (self.rect.x, self.rect.y))
```

Module Weapon.py :

+ Giải thích: nhằm gắn vũ khí lên người nhân vật, tạo ra góc quay cho vũ khí theo mũi tên con trỏ chuột. Khi nhấn chuột thì sẽ tương tác giữa cung và mũi tên bằng việc bắn mũi tên lao ra từ cung. Tạo sự tương tác giữa cung và kẻ thù bằng cách tạo sát thương gây ra từ mũi tên.

+ Docstring module Weapon:

_ Class Weapon:

Đại diện cho vũ khí của người chơi.

Attributes:

- original_image : Hình ảnh gốc của vũ khí.
- angle (float): Góc quay của vũ khí.
- image : Hình ảnh hiện tại của vũ khí sau khi được quay.
- arrow_image : Hình ảnh của mũi tên.
- rect :Hình chữ nhật bao quanh vũ khí.
- fired (bool): Xác định xem vũ khí đã được bắn ra hay chưa.
- last_shot (int): Thời điểm cuối cùng vũ khí được bắn ra.

Methods:

- update(player): Cập nhật trạng thái của vũ khí.
- draw(surface): Vẽ vũ khí lên màn hình.

_ Class Arrow:

Đại diện cho mũi tên trong trò chơi.

Attributes:

- original_image : Hình ảnh gốc của mũi tên.
- angle (float): Góc quay của mũi tên.
- image: Hình ảnh hiện tại của mũi tên sau khi được quay.
- rect : Hình chữ nhật bao quanh mũi tên.
- dx (float): Vận tốc theo phương ngang của mũi tên.
- dy (float): Vận tốc theo phương dọc của mũi tên.

Methods:

- update(screen_scroll, obstacle_titles, enemy_list): Cập nhật trạng thái của mũi tên.
- draw(surface): Vẽ mũi tên lên màn hình.

_ Class Fireball:

Đại diện cho quả cầu lửa trong trò chơi.

Attributes:

- original_image : Hình ảnh gốc của quả cầu lửa.
- angle (float): Góc quay của quả cầu lửa.
- image : Hình ảnh hiện tại của quả cầu lửa sau khi được quay.
- rect : Hình chữ nhật bao quanh quả cầu lửa.
- dx (float): Vận tốc theo phương ngang của quả cầu lửa.
- dy (float): Vận tốc theo phương dọc của quả cầu lửa.

Methods:

- update(screen_scroll, player): Cập nhật trạng thái của quả cầu lửa.
- draw(surface): Vẽ quả cầu lửa lên màn hình.

+ Code:

```
import random
import random
import pygame
import math
from constants import *

class Weapon():
```

```
    def __init__(self,image,arrow_image):
        self.original_image = image
        self.angle = 0
        self.image = pygame.transform.rotate(self.original_image,self.angle)
        self.arrow_image = arrow_image
        self.rect = self.image.get_rect()
        self.fired = False
        self.last_shot = pygame.time.get_ticks()
    def update(self,player):
        shot_cooldown = 300
        arrow = None
        self.rect.center = player.rect.center
        pos = pygame.mouse.get_pos()
        x_dist = pos[0] - self.rect.centerx
        y_dist = -(pos[1] - self.rect.centery)
        self.angle = math.degrees(math.atan2(y_dist,x_dist))

        #nhap chuot
        if pygame.mouse.get_pressed()[0] and self.fired == False and
(pygame.time.get_ticks()- self.last_shot) >= shot_cooldown:
            arrow =
Arrow(self.arrow_image,self.rect.centerx,self.rect.centery,self.angle)
            self.fired = True
            self.last_shot = pygame.time.get_ticks()
        if pygame.mouse.get_pressed()[0] == False:
            self.fired = False
        return arrow

    def draw(self,surface):
        self.image = pygame.transform.rotate(self.original_image,self.angle)
        surface.blit(self.image,((self.rect.centerx -
int(self.image.get_width()/2)) , self.rect.centery -
int(self.image.get_height()/2)))

class Arrow(pygame.sprite.Sprite):
    def __init__(self,image,x,y,angle):
        pygame.sprite.Sprite.__init__(self)
        self.original_image = image
        self.angle = angle
        self.image = pygame.transform.rotate(self.original_image,self.angle -
90)
        self.rect = self.image.get_rect()
        self.rect.center = (x,y)
        #tinh toan goc
        self.dx = math.cos(math.radians(self.angle)) * ARROW_SPEED
        self.dy = -(math.sin(math.radians(self.angle)) * ARROW_SPEED)
    def update(self,screen_scroll,obstacle_titles,enemy_list):
        #tao bien
        damage = 0
        damage_pos = None

        #dinh lai vi tri dua vao toc do
        self.rect.x += screen_scroll[0] + self.dx
        self.rect.y +=screen_scroll[1] +  self.dy
        #kiem tra collision voi buc tuong
        for obstacle in obstacle_titles:
            if obstacle[1].colliderect(self.rect):
```

```
                self.kill()

        #kiem tra mui ten di ra khoi map hay chua
        if self.rect.right < 0 or self.rect.left >  SCREEN_WIDTH or
self.rect.bottom < 0 or self.rect.top > SCREEN_HEIGHT:
            self.kill()
        #kiem tra mui ten dinh ke dich hay khong ?
        for enemy in enemy_list:
            if enemy.rect.colliderect(self.rect) and enemy.alive:
                damage = 10 + random.randint(-5,5)
                damage_pos = enemy.rect
                enemy.health -= damage
                enemy.hit = True
                self.kill()
                break

        return damage,damage_pos
    def draw(self,surface):
        surface.blit(self.image, ((self.rect.centerx -
int(self.image.get_width() / 2)), self.rect.centery -
int(self.image.get_height() / 2)))


class Fireball(pygame.sprite.Sprite):
    def __init__(self,image,x,y,target_x,target_y):
        pygame.sprite.Sprite.__init__(self)
        self.original_image = image
        x_dist = target_x - x
        y_dist = -(target_y - y)
        self.angle = math.degrees(math.atan2(y_dist,x_dist))
        self.image = pygame.transform.rotate(self.original_image,self.angle -
90)
        self.rect = self.image.get_rect()
        self.rect.center = (x,y)

        #tinh toan van toc theo phuong thang dung hoac nam ngang dua vao goc
        self.dx = math.cos(math.radians(self.angle)) * FIREBALL_SPEED
        self.dy = -(math.sin(math.radians(self.angle)) * FIREBALL_SPEED)
    def update(self,screen_scroll,player):
        #reset bien
        damage = 0
        damage_pos = None

        #dinh lai vi tri dua vao toc do
        self.rect.x += screen_scroll[0] + self.dx
        self.rect.y +=screen_scroll[1] +  self.dy

        #kiem tra fireball di ra khoi map hay chua
        if self.rect.right < 0 or self.rect.left >  SCREEN_WIDTH or
self.rect.bottom < 0 or self.rect.top > SCREEN_HEIGHT:
            self.kill()
        #kiem tra o giua player va doi tuong co buc tuong khong
        if player.rect.colliderect(self.rect) and player.hit == False:
            player.hit = True
            player.last_hit = pygame.time.get_ticks()
            player.health -=10
            self.kill()
```

```
    def draw(self,surface):
        surface.blit(self.image, ((self.rect.centerx -
int(self.image.get_width() / 2)), self.rect.centery -
int(self.image.get_height() / 2)))
```

Module items.py:

+ Giải thích : chủ yếu để xử lý 2 item chính của trò chơi là đồng xu và thuốc hồi máu, đồng xu animate (được tạo từ 4 bức ảnh) và 1 hình ảnh lọ thuốc hồi máu. Class Item chủ yếu tập trung cho việc tương tác giữa tác nhân và 2 item chính , cộng thêm 1 số chức năng là tính điểm và hồi máu cho nhân vật.

+ Docstring module Item :

_ Class Items :

Đại diện cho các vật phẩm trong trò chơi.

Attributes:

- item_type (int): Loại của vật phẩm (0: đồng xu, 1: hồi máu).
- animation_list (list): Danh sách các hình ảnh tạo nên hoạt ảnh của vật phẩm.
- frame_index (int): Chỉ số của hình ảnh hiện tại trong danh sách hoạt ảnh.
- update_time (int): Thời điểm cập nhật cuối cùng của hoạt ảnh.
- image : Hoạt ảnh của hình ảnh hiện tại của vật phẩm tại frame_index nhất định
- rect(pygame.rect) : Hình chữ nhật bao quanh vật phẩm.
- dummy_coin (bool): Xác định xem vật phẩm có phải là đồng xu giả mạo hay không.

Methods:

- update(screen_scroll, player): Cập nhật trạng thái của vật phẩm, đồng thời có hiệu ứng khi sound effect khi vật phẩm nhặt lên.
- draw(surface): Vẽ vật phẩm lên màn hình.

+ Code :

```
import pygame
class Item(pygame.sprite.Sprite):
    def __init__(self,x,y,item_type,animation_list,dummy_coin = False):
        pygame.sprite.Sprite.__init__(self)
        self.item_type = item_type
        self.animation_list = animation_list
        self.frame_index = 0
        self.update_time = pygame.time.get_ticks()
        self.image = self.animation_list[self.frame_index]
        self.rect = self.image.get_rect()
        self.rect.center = (x,y)
        self.dummy_coin = dummy_coin
    def update(self,screen_scroll,player,coin_fx,heal_fx):
        #dummy coin
        if not self.dummy_coin:
            self.rect.x += screen_scroll[0]
            self.rect.y += screen_scroll[1]
        #kiem tra xem nhan vat lay item hay chua
        if self.rect.colliderect(player.rect):
            #thu thap coin
            if self.item_type == 0:
                player.score += 1
                coin_fx.play()
            elif self.item_type == 1:
                player.health += 10
                heal_fx.play()
                if player.health > 100:
                    player.health = 100
            self.kill()

        #hoat anh cho item
        animation_cooldown = 150
        self.image = self.animation_list[self.frame_index]
        if pygame.time.get_ticks() - self.update_time > animation_cooldown:
            self.frame_index +=1
            self.update_time = pygame.time.get_ticks()
        if self.frame_index >= len(self.animation_list):
            self.frame_index = 0

    def draw(self,surface):
        surface.blit(self.image,self.rect)
```

Module world.py:

+ Giải thích : Xử lý cái map bằng cách xử lý rời rạc từng pixel trên map bằng cách xử lý các thứ tự ví dụ 1,2,3,… tương ứng với các tác nhân trên bản đồ. Các tác nhân được sắp xếp theo thứ tự 1,2,3,… tạo thành list để xử lý khi đưa thư mục csv vào.

+ Docstring module world:

- Class World

Đại diện cho thế giới trong trò chơi và các phương thức quản lý.

Attributes:

- map_titles (list): Danh sách các ô trên bản đồ.
- obstacle_tiles (list): Danh sách các ô chướng ngại vật trên bản đồ.
- exit_tile (list): Thông tin về ô đích ra khỏi bản đồ.
- item_list (list): Danh sách các vật phẩm trên bản đồ.
- player (class Character): Thông tin về nhân vật người chơi.
- character_list (list): Danh sách các nhân vật khác trên bản đồ.

Methods:

- process_data(data, tile_list, item_images, mob_animations): Xử lý dữ liệu bản đồ và tạo các đối tượng.
- update(screen_scroll): Cập nhật vị trí của các ô trên bản đồ dựa trên việc di chuyển màn hình.
- draw(surface): Vẽ các ô trên bản đồ lên màn hình trò chơi.

+ Code :

```
from character import character
from items import Item
from constants import *
class World():
    def __init__(self):
        self.map_titles = []
        self.obstacle_tiles = []
        self.exit_tile = None
        self.item_list = []
        self.player = None
        self.character_list =[]

    def process_data(self,data,tile_list,item_images,mob_animations):
        self.level_length = len(data)
        #lap lai tung data trong file data
        for y,row in enumerate(data):
            for x,tile in enumerate(row):
                image = tile_list[tile]
                image_rect = image.get_rect()
                image_x = x * TILE_SIZE
```

```
                image_y = y * TILE_SIZE
                image_rect.center = (image_x,image_y)
                tile_data = [image,image_rect,image_x,image_y]
                if tile == 7:
                    self.obstacle_tiles.append(tile_data)
                elif tile == 8:
                    self.exit_tile = tile_data
                elif tile ==9 :
                    coin = Item(image_x,image_y,0,item_images[0])
                    self.item_list.append(coin)
                    tile_data[0] = tile_list[0]
                elif tile ==10:
                    potion = Item(image_x,image_y,1,[item_images[1]])
                    self.item_list.append(potion)
                    tile_data[0] = tile_list[0]
                elif tile ==11 :
                    player =
character(image_x,image_y,100,mob_animations,0,False,1.25)
                    self.player = player
                    tile_data[0] = tile_list[0]
                elif tile >= 12 and tile <= 16:
                    enemy = character(image_x,image_y,100,mob_animations,tile
-11,False,1)
                    self.character_list.append(enemy)
                    tile_data[0] = tile_list[0]
                elif tile == 17:
                    enemy =
character(image_x,image_y,100,mob_animations,6,True,2.5)
                    self.character_list.append(enemy)
                    tile_data[0] = tile_list[0]
                # dua du lieu tu image data vao tile list
                if tile >=0:
                    self.map_titles.append(tile_data)


    def update(self,screen_scroll):
        for tile in self.map_titles:
            tile[2] += screen_scroll[0]
            tile[3] += screen_scroll[1]
            tile[1].center = (tile[2],tile[3])
    def draw(self,surface):
        for tile in self.map_titles:
            surface.blit(tile[0],tile[1]) #0 load nhan vat, #1 load pixel map
```

Module button.py:

+ Giải thích: Tạo ra các hình ảnh MENU như START,RESUME,EXIT,… để người dung có thể bật game , tắt game, tạm dừng trò chơi và khi thua sẽ hiển thị chơi lại.

+ Docstring module button:

Lớp đại diện cho hình ảnh MENU trong trò chơi.

Attributes:

x (int): Tọa độ x của nút trên màn hình.

y (int): Tọa độ y của nút trên màn hình.

image (pygame.Surface): Hình ảnh của nút.

rect (pygame.Rect): Hình chữ nhật bao quanh nút.

Methods:

draw(surface): Vẽ nút lên màn hình và kiểm tra xem các nút Start,Resume,.. có được nhấn hay không.

+ Code :

```
import pygame
class Button():
    def __init__(self,x,y,image):
        self.image = image
        self.rect = self.image.get_rect()
        self.rect.topleft =(x,y)

    def draw(self,surface):
        action = False
        #di chuyen con chuot
        pos = pygame.mouse.get_pos()

        #click chuot
        if self.rect.collidepoint(pos):
            if pygame.mouse.get_pressed()[0]:
                action = True

        surface.blit(self.image,self.rect)
        return action
```

Module thông số Constants.py:

+ Giải thích: Một số tham số để scale nhân vật , vận tốc mũi tên và khung hình window pygame

```
FPS = 60
SCREEN_WIDTH = 800
SCREEN_HEIGHT = 600

SCALE = 2.5
SCAlE_GOBLIN = 1.3
SCALE_MONSTER = 1.4
SCALE_STEAL = 1.2
```

```
WEAPON_SCALE = 1.5
ITEM_SCALE = 3
POTION_SCALE = 2
FIREBALL_SCALE = 1
BUTTON_SCALE = 1

FIREBALL_SPEED = 7
SPEED = 5
ARROW_SPEED = 10
ENEMY_SPEED = 4
OFFSET = 5
TILE_SIZE = 16 * SCALE
TILE_TYPES = 18

ROW = 150
COLS = 150
SCROLL_THRESH = 200
RANGE = 50
ATTACK_RANGE = 60

PANEL = (50,50,50)
WHITE = (255,255,255)
BLACK = (0,0,0)
PINK = (235,65,54)
RED = (255,0,0)
BG = (40,25,25)
MENU_BG = (0,0,30)
```