TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN
MÔN LẬP TRÌNH PYTHON

TÊN SINH VIÊN: **TRẦN DƯƠNG ĐẮC LỘC - 3121410304**

TÊN ĐỀ TÀI: **XÂY DỰNG TRÒ CHƠI BẮN SÚNG 2D VỚI THƯ VIỆN PYGAME BẰNG NGÔN NGỮ LẬP TRÌNH PYTHON**

TP.HCM tháng 5/2023

NHẬN XÉT, ĐÁNH GIÁ CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn Khoa Công nghệ thông tin, trường Đại học Sài Gòn đã tạo điều kiện cho chúng em được thực hiện đồ án môn học "Ngôn ngữ lập trình Python". Chúng em cũng xin chân thành cảm ơn thầy Trịnh Tấn Đạt, thầy đã giảng dạy cho chúng em những kiến thức cần thiết cho môn học này. Những kiến thức đó đã giúp cho chúng em rất nhiều trong quá trình làm đồ án báo cáo môn học. Chúng em xin chân thành cám ơn quý Thầy cô trong Khoa đã tận tình giảng dạy và trang bị cho chúng em những kiến thức cần thiết trong thời gian qua. Mặc dù nhóm đã cố gắng hoàn thành đồ án môn học với tất cả nổ lực của từng thành viên trong nhóm, nhưng đồ án chắc chắn không tránh khỏi nhữmg thiếu sót nhất định, rất mong nhận được sự cảm thông, chia sẻ và tận tình đóng góp chỉ bảo của quý Thầy Cô.

Chúng em xin chân thành cảm ơn thầy.

A.Mở Đầu

1. Lý do chọn đề tài

Hiện nay, ngành Công nghệ thông tin đã và đang có những bước phát triển nhanh chóng ứng dụng của nó trong mọi lĩnh vực trong cuộc sống trên thế giới nói chung và Việt Nam nói riêng. Công nghệ thông tin là một phần không thể thiếu của cuộc sống, góp phần đẩy mạnh công cuộc công nghiệp hóa, hiện đại hóa và quá trình phát triển của đất nước. Việc ứng dụng những thành quả của khoa học công nghệ vào trong đời sống, trong công tác là hết sức thiết yếu. Ứng dụng của công nghệ thông tin kết hợp với truyền thông hóa được xem là một trong những yếu tố mang tính quyết định trong mọi hoạt động của các công ty, tổ chức... nó đóng vai trò quan trọng và không thể thiếu. Công nghệ thông tin và truyền thông hóa góp phần làm thay đổi suy nghĩ, lối mòn tư duy của mỗi con người, nó giúp con người năng động hơn, kết nối nhanh hơn ở moi lúc mọi nơi làm tăng hiệu quả và năng suất của công việc.

Python là ngôn ngữ lập trình hướng đối tượng bậc cao, dùng để phát triển website và nhiều ứng dụng khác nhau. Python được tạo ra bởi Guido van Rossum và được phát triển trong một dự án mã mở (open source). Với cú pháp cực kì đơn giản và thanh lịch, Python là lựa chọn hoàn hảo cho những ai lần đầu tiên học lập trình. cho nên có thể được dùng để tạo ra các ứng dụng để giải quyết các vấn đề về số, xử lý văn bản, tạo ra trò chơi, và nhiều thứ khác.

Trong quá trình tìm hiểu chúng em thấy rất hứng thú với các ứng dụng game được cài đặt và lập trình bằng ngôn ngữ lập trình Python bằng thư viện Pygame. Pygame là một bộ công cụ tiện ích, là một phần của ngôn ngữ lập trình Python, có rất nhiều tựa game huyền thoại đời đầu đã được tạo nên từ ngôn ngữ Python, nên chúng em đã quyết định sử dụng thư viện Pygame của Python để xây dựng một game bắn súng 2d cơ bản để thấy rõ được khả năng mạnh mẽ của nó.

2. Mục đích – mục tiêu của đề tài

a. Mục đích:
- Nắm được kỹ năng và kiến thức cơ bản về lập trình game
- Tìm hiểu về thư viện pygame trong python
- Áp dụng kiến thức từ 2 mục đích trên vào thực tế

b. Mục tiêu:
- Thấy được khả năng của pygame trong lập trình game

B. Xây Dựng Trò Chơi Bắn Súng 2d Với Thư Viện Pygame

1. Giới thiệu về game bắn súng 2d

Game bắn súng là thể loại phụ của game hành động, trong đó game bắn súng 2d hay game chạy bắn là thể loại game mà màn hình của người chơi sẽ có thể di chuyển qua trái hoặc phải (hoặc có thể lên xuống) để di chuyển nhân vật đến các vị trí trên bản đồ. Di chuyển nhân vật vừa đi vừa bắn để tiêu diệt các mục tiêu và hoàn thành nhiệm vụ.

2. Ý tưởng

a. Các đối tượng

- Nhân vật: nhân vật bao gồm người chơi và quân địch. Người chơi và quân địch sử dụng vũ khí để tiêu diệt nhau
- Bối cảnh: bối cảnh là 1 khu rừng, nhân vật phải tiêu diệt kẻ thù để tiến vào sâu hơn để hoàn thành màn chơi (map)
- Map: sử dụng cấu trúc Tilemap là một mạng lưới các ô gạch, mỗi một ô là một hình ảnh tương ứng với bối cảnh (ô chứa vật cản, ô xuất hiện quân địch, ô đích đến, ô người chơi bắt đầu,...)
- Vũ khí: súng, bom
- Vật phẩm: đạn và bom nhặt để tăng số lượng
- Vật cản: tường, đất, nước... để tạo bản đồ

b. Tính năng

- Di chuyển: di chuyển nhân vật qua trái qua phải, nhảy có trọng lực
- Tấn công: nhân vật có thể sử dụng vũ khí súng để bắn và ném bom
- Nhảy có áp dụng trọng lực để né và di chuyển
- Có thể tự tạo map dễ dàng

3. Chi tiết code

a. Các biến và hàm toàn cục

SCREEN_WIDTH = 800	Độ dài màn hình, giá trị mặc định là 800
SCREEN_HEIGHT	Độ rộng màn hình, giá trị mặc định sẽ bằng SCREEN_WIDTH * 0.8
screen	Lưu trữ thông tin màn hình game dưới dạng Surface object của

	pygame
Clock	Lưu trữ thông tin hệ thống đồng hồ trong game dưới dạng Clock object của pygame
FPS	Số khung hình (hình ảnh) mà game xuất ra màn hình trong một giây
GRAVITY	Độ lớn của trọng lực áp dụng lên nhân vật trong game
JUMP_VEL	Độ lớn của lực nhảy của nhân vật trong game
SCROLL_THRESH	Khi khoảng cách giữa nhân vật và màn hình cách 1 khoảng SCROLL_THRESH thì màn hình sẽ di chuyển theo nhân vật
ROWS	Số hàng tối đa của map để xây dựng cấu trúc Tilemap
COLS	Số cột tối đa của map để xây dựng cấu trúc Tilemap
TILE_SIZE	Độ rộng của mỗi ô vuông trong Tilemap
TILE_TYPES	Số lượng loại hình ảnh được sử dụng trong mỗi ô vuông của Tilemap
MAX_LEVELS	Số level tối đa hiện tại của game
screen_scroll	Lưu trữ độ dài chênh lệch mà màn hình đã di chuyển so với vị trí ban đầu
bg_scroll	Lưu trữ độ dài chênh lệch mà hình nền trong map đã di chuyển so với vị trí ban đầu
Level	Level đầu bắt khi vừa vào game
start_game	Biến trạng thái cho biết đã vào map hay chưa
start_intro	Biến trạng thái cho biết đang load màn hình chuyển cảnh
moving_left = False moving_right = False shoot = False grenade = False grenade_thrown = False	Các biến trạng thái input từ bàn phím để điều khiển nhân vật
*_fx	Các biến lưu trữ âm thanh (* tương ứng với ký tự bất kì)

*_img	Các biến lưu trữ hình ảnh các vật thể trong game
Img_list	Mảng lưu các hình ảnh sử dụng để xây dựng map (vd: vật cản, nước,...)
BG = (144, 201, 120) RED = (255, 0, 0) WHITE = (255, 255, 255) GREEN = (0, 255, 0) BLACK = (0, 0, 0) PINK = (235, 65, 54)	Định nghĩa màu sắc
font	Phông chữ dùng trong game
draw_text()	Vẽ chữ lên màn hình
draw_bg()	Vẽ ảnh lên màn hình
reset_level()	Khởi tạo lại các biến dùng để tạo map
intro_fade	Màn hình chuyển cảnh khi vào map
death_fade	Màn hình chuyển cảnh khi chết
start_button	Nút bắt đầu vào map
exit_button	Nút thoát game
restart_button	Nút bắt đầu lại khi chết
*_group	Các lớp sprite.Group chứa các sprite vật thể
world_data	Dữ liệu map đọc từ file văn bản level..._data.csv
world	Class chứa dữ liệu map đã được chuyển đổi từ dữ liệu văn bản thành dữ liệu vật thể trong game
player, health_bar	Người chơi và thanh máu được tải từ hàm process_data() của world
run	Trạng thái của game đang chạy hoặc không

b. Nhân vật

-Lớp (class) Soldier được định nghĩa là lớp nhân vật trong game, bao gồm cả người chơi và quân địch. Lớp này có kế thừa lớp Sprite của pygame (pygame.sprite.Sprite)

-Các thuộc tính và hàm:

Alive	true hoặc false thể hiện trạng thái nhân vật sống hoặc chết
Char_type	cho biết đây là người chơi hay là quân địch
Speed	là tốc độ di chuyển theo trục x của nhân vật (trái và phải)
X, Y	là vị trí theo trục toạ độ 0xy của nhân vật khi vừa được khởi tạo
Ammo	lượng đạn mà nhân vật có khi được khởi tạo
Shoot_cooldown	thời gian cần đợi cho mỗi lần bắn
Grenades	lượng lựu đạn (bom) mà nhân vật có khi được khởi tạod
Health	lượng máu của nhân vật
Max_health	lượng máu cao nhất của nhân vật, dùng để ngăn các vật phẩm không hồi quá mức này
Direction	1 và 0 tương ứng hướng mặt của nhân vật theo hướng phải và trái
Vel_y	là vận tốc theo chiều dọc của nhân vật, dùng để theo dõi trạng thái nhảy hoặc rơi của nhân vật
Jump	true hoặc false tương ứng trạng thái đang nhảy hoặc không
in_air	true hoặc false tương ứng trạng thái đang trên không hoặc ở mặt đất của nhân vật
flip	true hoặc false trạng thái lật của hình ảnh nhân vật, dùng để lật ảnh theo chiều dọc theo hướng di chuyển
animation_list	danh sách 2 chiều, mỗi 1 giá trị tương ứng là loại hành động và mỗi loại hành động là 1 mảng các hình ảnh dùng cho chuyển động của nhân vật.
frame_index	số thứ tự hình ảnh mà nhân vật đang được thể hiện trên màn hình
action	số thứ tự hành động của nhân vật
update_time	sử dụng function pygame.time.get_ticks() để lấy thời gian tối đa của mỗi khung hình game
move_counter (biến sử dụng cho AI quân địch)	số bước chân tối đa mà nhân vật AI di chuyển được trước khi chuyển hướng
Scale	là tỉ lệ kích thước của nhân vật được khởi tạo so với kích thước hình ảnh được dùng làm nhận dạng của nhân vật

__init__()	Cài đặt các thuộc tính mặc định
update()	Được gọi mỗi lần khung hình trò chơi cập nhật
move()	Nhận 2 đầu vào moving_left và moving_right, tương ứng với mỗi giá trị khác 0 sẽ di chuyển nhân vật theo hướng trái và phải. Ngoài ra còn kiểm tra: - Bước đi tiếp theo có va chạm vật thể hay không (sử dụng colliderect và spritecollide): o Va chạm chiều trục Ox: với người chơi sẽ đứng yên tại chỗ, với quân địch thì sẽ di chuyển ngược lại o Va chạm chiều trục Oy: nếu chạm vật thể sẽ cho bước đi tiếp theo tối đa bằng khoảng cách giữa vật thể và nhân vật o Va chạm nước: nhân vật chết o Chạm đích: hoàn thành map o Ra khỏi map: nhân vật chết o Chạm 2 góc màn hình: người chơi sẽ đứng yên

	- Di chuyển màn hình nhân vật khi nhân vật cách SCROLL_THRESH so với cạnh màn hình - Trả về giá trị screen_scroll và level_complete tương ứng là độ dài màn hình bị kéo đi so với map và biến cho biết đã chạm đến đích hay chưa
shoot()	Kiểm tra nếu còn đạn và thời gian đợi mỗi lần bắn bằng 0 thì sẽ tạo vật thể Bullet và đưa bullet vào danh sách vật thể bullet dùng cho check va chạm đạn
ai()	Hàm tạo thao tác tự động cho nhân vật quân địch, nếu quân địch và nhân vật còn sống: - Tạo hành động ngẫu nhiên đứng yên với tỉ lệ 1/200 nếu hành động hiện tại của quân địch không phải đứng yên với thời gian là idling_counter - Kiểm tra tầm nhìn của quân địch có người chơi: sẽ đứng lại và bắn người chơi - Nếu không trong tầm nhìn: thì quân địch sẽ ngẫu nhiên di chuyển trong tầm TILE_SIZE và sẽ di chuyển ngược lại khi đạt tầm tối đa Cập nhật toạ độ Ox của quân địch khi màn hình di chuyển theo biến toàn cục screen_scroll
update_animation()	Cập nhật hình ảnh chuyển động nhân vật, sau mỗi ANIMATION_COOLDOWN sẽ thay đổi ảnh hiện tại của nhân vật trong phạm vi animation_list theo action hiện tại và
update_action()	Cập nhật hành động của nhân vật khi hành động của nhân vật thay đổi so với hiện tại
check_alive()	Kiểm tra trạng thái sự sống của nhân vật
draw()	Vẽ nhân vật lên màn hình

c. Bối cảnh:

-Class World được định nghĩa để đọc, lưu trữ và vẽ các vật thể trong map, kế thừa lớp sprite của pygame

-Các thuộc tính và hàm:

obstacle_list	Danh sách các trở ngại
process_data()	Đọc dữ liệu từ file định nghĩa level map và tạo các vật thể trong game theo toạ độ được định sẵn trong file Với mỗi ô dữ liệu trong file dữ liệu map sẽ lưu 1 giá trị số đặt là Tile: 0 <= Tile <= 8: Vật cản, mặt đất 9 <= Tile <= 10: Vật thể nước 11 <= Tile <= 14: Vật thể trang trí Tile = 15: Người chơi Tile = 16: Quân địch Tile = 17: vật phẩm nhặt đạn Tile = 18: vật phẩm nhặt lựu đạn Tile = 19: vật phẩm nhặt hồi máu Tile = 20: vật phẩm đích hoàn thành map

	Trả về 2 giá trị là player và healthbar tương ứng là 2 đối tượng Soldier và HealthBar đã đọc và tạo trong hàm
draw()	Vẽ các đối tượng vật cản trong obstacle_list lên màn hình

d. Các vật thể khác

-Class Decoration được định nghĩa để lưu các vật thể trang trí trong map.

-Class Wather được định nghĩa là đối tượng nước trong game, khi người chơi chạm vào nước sẽ chết

-Class Exit được định nghĩa là đối tượng hoàn thành map trong game, khi người chơi chạm vào sẽ được đến màn tiếp theo

-Kế thừa lớp sprite của pygame

-Các thuộc tính và hàm:

img	Lưu hình ảnh của vật thể
x, y	Lưu toạ độ trục Oxy của vật thể
update()	Cập nhật toạ độ của vật thể khi màn hình bị kéo đi theo biến toàn cục screen_scroll

e. Các vật phẩm nhặt

-Class ItemBox được định nghĩa là các vật phẩm có thể nhặt được trong game

-Kế thừa lớp sprite của pygame

-Các thuộc tính và hàm:

item_type	Loại vật phẩm
x, y	Toạ độ trục Oxy của vật thể
update()	Cập nhật trạng thái đã bị nhặt hay chưa, nếu bị nhặt sẽ xử lý tuỳ vào item_type và sau đó xoá vật thể bằng hàm kill()

f. Thanh máu

-Class HealthBar được định nghĩa để lưu các thông tin về thanh máu của nhân vật

-Các thuộc tính và hàm:

x, y	Toạ độ trục Oxy của vật thể
health	Lượng máu hiện tại
max_health	Lượng máu tối đa
draw()	Tính toán và thanh máu lên màn hình

g. Đạn

-Class Bullet được dùng để định nghĩa vật thể đạn sau khi súng bắn ra

-Kế thừa lớp sprite của pygame

-Các thuộc tính và hàm:

x, y	Toạ độ trục Oxy của vật thể
direction	Hướng di chuyển của đạn
update()	Di chuyển viên đạn thẳng theo hướng trái hoặc phải Kiểm tra nếu đạn đã ra ngoài màn hình hoặc chạm vật thể khác sẽ xoá bản thân viên đạn Kiểm tra nếu đạn chạm vào nhân vật sẽ trừ máu của nhân vật

h. Lựu đạn

-Class Grenade được dùng để định nghĩa vật thể lựu đạn sau khi ném ra

-Kế thừa lớp sprite của pygame

-Các thuộc tính và hàm:

x, y	Toạ độ trục Oxy của vật thể

direction	Hướng di chuyển của vật thể
update()	Di chuyển lựu đạn theo hình vòng cung và áp dụng trọng lực lên vật thể Xử lý va chạm với vật cản Cập nhật thời gian đợi giữa mỗi lần ném lựu đạn

i. Hoạt ảnh và xử lý lựu đạn nổ

-Class Explosion được dùng để định nghĩa hoạt ảnh và xử lý nổ của
-Kế thừa lớp sprite của pygame
-Các thuộc tính và hàm:

x, y	Toạ độ trục Oxy của vật thể
scale	Tỉ lệ của hoạt ảnh nổ
update()	Cập nhật toạ độ của vật thể khi màn hình bị kéo đi theo biến toàn cục screen_scroll Cập nhật hoạt ảnh nổ theo thời gian

j. Màn hình chuyển cảnh

- Class ScreenFade được dùng để định nghĩa hoạt ảnh chuyển cảnh giữa các màn hình game (khi vào map, kết thúc map, chết,...)
- Các thuộc tính và hàm:

direction	Hướng di chuyển của màn hình chuyển cảnh
color	Màu sắc của màn hình chuyển cảnh
speed	Tốc độ của màn hình chuyển cảnh
fade()	Vẽ màn hình chuyển cảnh theo direction 1. Mở từ trong ra ngoài 2. Mở theo chiều ngang

k. Chương trình chính

```
while run:

    clock.tick(FPS)
```

- Là một vòng lặp while dùng để để cập khung hình mới theo thời gian
- clock.tick(FPS): hàm kiểm tra đã trải qua bao nhiêu mili giây kể từ lần gọi trước đó, truyền vào tham số FPS sẽ làm cho hàm tự động delay (trì hoãn) game 1 khoảng thời gian nếu thời gian gọi trước đó nhỏ hơn thời gian mỗi khung hình để mỗi khung hình sẽ được cập nhật theo đúng thời gian đã đặt. Đặt FPS sẽ giúp thời gian và các tính toán trong game chính xác hơn khi chúng ta tính toán dựa trên mỗi lần khung hình cập nhật
- Kiểm tra nếu chưa bắt đầu (Đang ở màn hình chính):

```
#draw menu
screen.fill(BG)
#add buttons
if start_button.draw(screen):
    start_game = True
    start_intro = True
if exit_button.draw(screen):
    run = False
```

+ Vẽ màn hình chính và xử lý sự kiện nhấn

+ Vẽ 2 nút bắt đầu và xử lý sự kiện nhấn

- Nếu game đã bắt đầu:

```
#update background
draw_bg()
#draw world map
world.draw()
#show player health
health_bar.draw(player.health)
#show ammo
draw_text('AMMO: ', font, WHITE, 10, 35)
for x in range(player.ammo):
    screen.blit(bullet_img, (90 + (x * 10), 40))
#show grenades
draw_text('GRENADES: ', font, WHITE, 10, 60)
for x in range(player.grenades):
    screen.blit(grenade_img, (135 + (x * 15), 60))
```

- Cập nhật hình ảnh nền của map
- Cập nhật các vật cản của map
- Cập nhật thanh máu của người chơi
- Vẽ số lượng đạn và lựu đạn của người chơi

```
player.update()
player.draw()

for enemy in enemy_group:
    enemy.ai()
    enemy.update()
    enemy.draw()

#update and draw groups
bullet_group.update()
grenade_group.update()
explosion_group.update()
item_box_group.update()
decoration_group.update()
water_group.update()
exit_group.update()
bullet_group.draw(screen)
grenade_group.draw(screen)
explosion_group.draw(screen)
item_box_group.draw(screen)
decoration_group.draw(screen)
water_group.draw(screen)
exit_group.draw(screen)
```

- Gọi các hàm cập nhật và vẽ của các đối tượng vật thể trong map

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

- Kiểm tra trạng thái chuyển cảnh và chuyển cảnh

```
if player.alive:
    #shoot bullets
    if shoot:
        player.shoot()
    #throw grenades
    elif grenade and grenade_thrown == False and player.grenades > 0:
        grenade = Grenade(player.rect.centerx + (0.5 * player.rect.size[0] * player.direction),\
                    player.rect.top, player.direction)
        grenade_group.add(grenade)
        #reduce grenades
        player.grenades -= 1
        grenade_thrown = True
    if player.in_air:
        player.update_action(2)#2: jump
    elif moving_left or moving_right:
        player.update_action(1)#1: run
    else:
        player.update_action(0)#0: idle
    screen_scroll, level_complete = player.move(moving_left, moving_right)
    bg_scroll -= screen_scroll
    #check if player has completed the level
    if level_complete:
        start_intro = True
        level += 1
        bg_scroll = 0
        world_data = reset_level()
        if level <= MAX_LEVELS:
            #load in level data and create world
            with open(f'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()
            player, health_bar = world.process_data(world_data)
```

- Xử lý khung hình khi nhân vật còn sống

```
screen_scroll = 0
if death_fade.fade():
    if restart_button.draw(screen):
        death_fade.fade_counter = 0
        start_intro = True
        bg_scroll = 0
        world_data = reset_level()
        #load in level data and create world
        with open(f'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()
        player, health_bar = world.process_data(world_data)
```

- Xử lý khung hình khi nhân vật chết

```
for event in pygame.event.get():
    #quit game
    if event.type == pygame.QUIT:
        run = False
    #keyboard presses
    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_SPACE:
            shoot = True
        if event.key == pygame.K_q:
            grenade = True
        if event.key == pygame.K_w and player.alive:
            player.jump = True
            jump_fx.play()
        if event.key == pygame.K_ESCAPE:
            run = False
```

- Xử lý sự kiện nhấn bàn phím
  + A: di chuyển sang trái
  + B: di chuyển sang phải
  + Space: bắn
  + Q: quăng lựu đạn
  + W: nhảy
  + ESC: thoát game

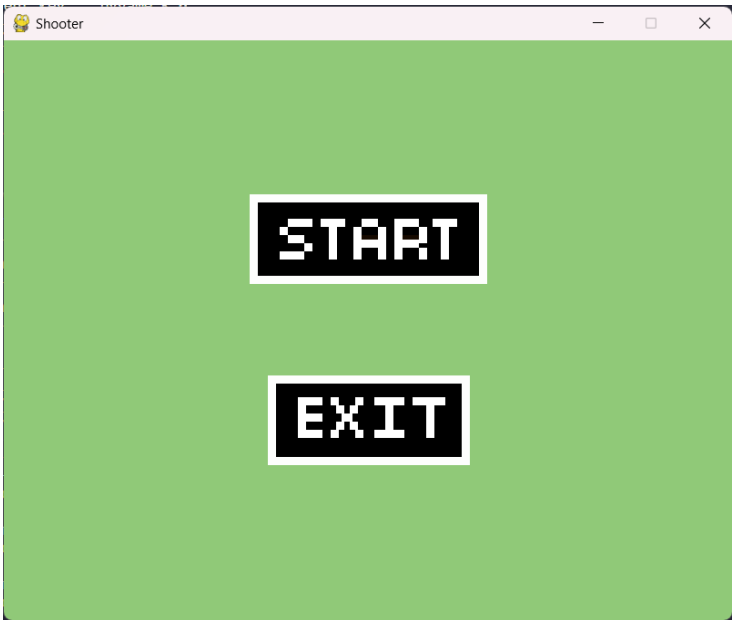
```
#keyboard button released
if event.type == pygame.KEYUP:
    if event.key == pygame.K_a:
        moving_left = False
    if event.key == pygame.K_d:
        moving_right = False
    if event.key == pygame.K_SPACE:
        shoot = False
    if event.key == pygame.K_q:
        grenade = False
        grenade_thrown = False
```

- Xử lý sự kiện thả phím (nhấc tay ra khỏi phím):

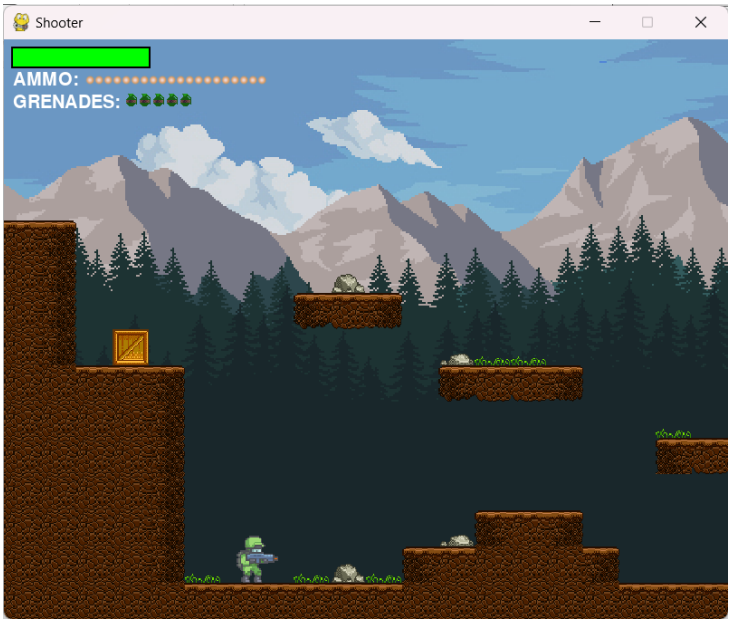
```
pygame.display.update()
```

- Cập nhật lại màn hình chương trình và kết thúc vòng lặp chương trình chính

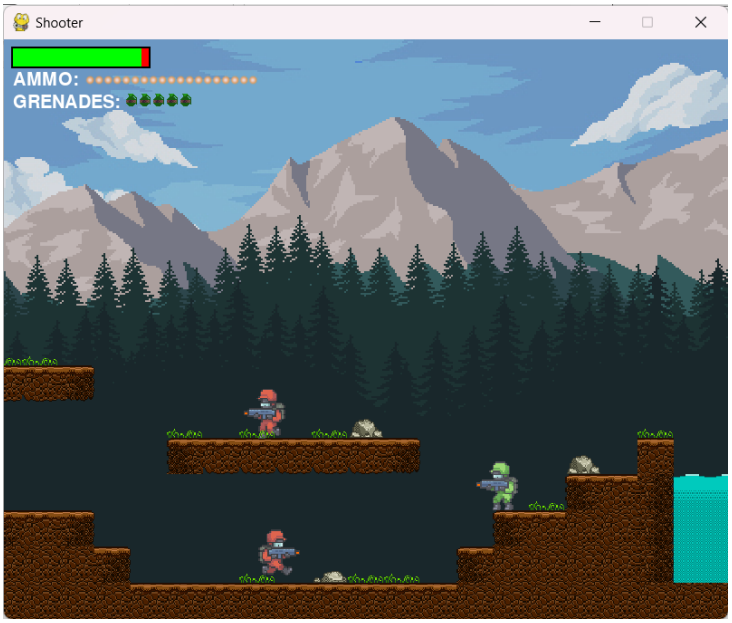
C. Demo Game



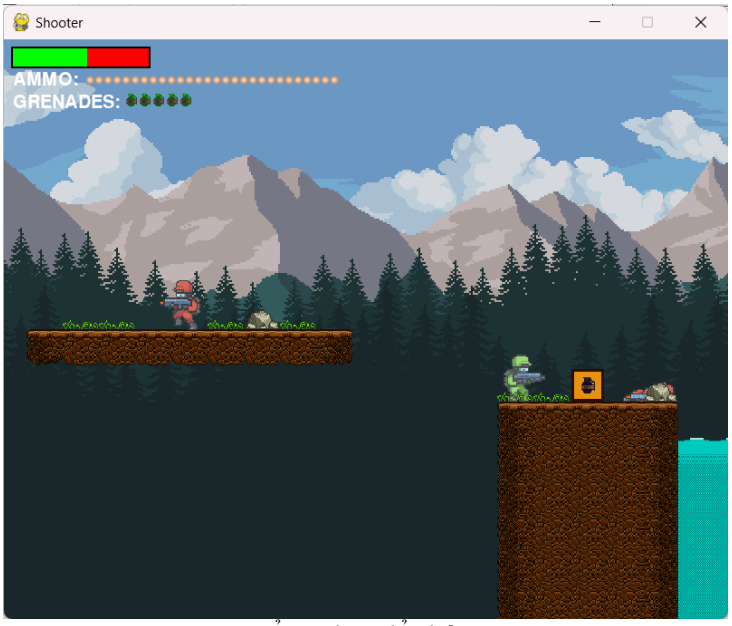
Màn hình chính



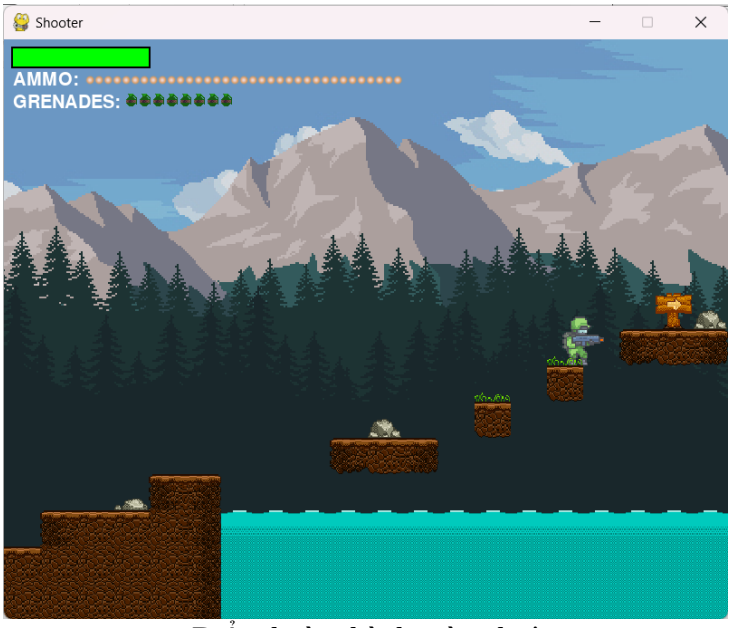
Màn hình trong game



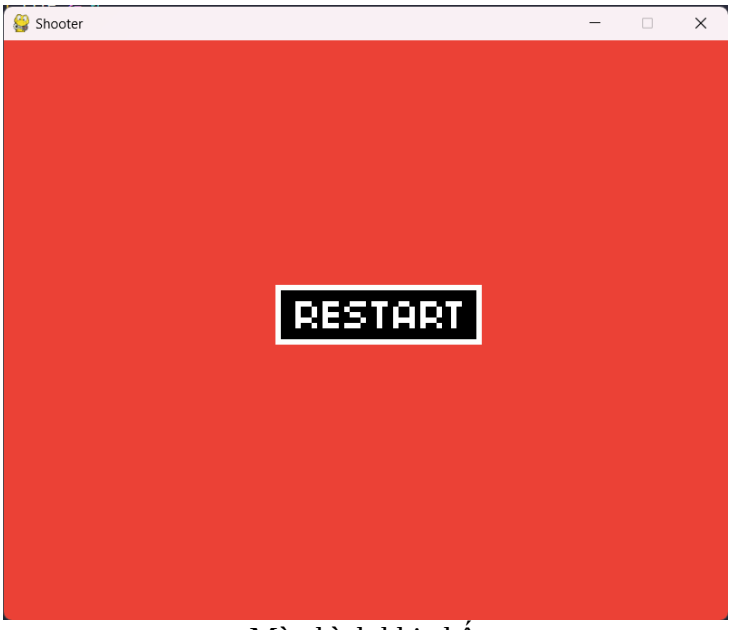
Các quân địch



Vật phẩm nhặt để thêm bom



Điểm hoàn thành màn chơi



Màn hình khi chết

Các tài liệu tham khảo:

https://github.com/russs123/Shooter
https://www.pygame.org/docs/