SPACE INVADERS

Báo cáo môn IT – nhóm 01CLC

Thành viên nhóm:
Hoàng Đoàn Tiến Phát - 20146266
Huỳnh Thiên Phúc - 20146267
Lê Huy Anh - 20146227

I. Các câu lệnh được sử dụng

I. Các câu lệnh được sử dụng

```
import pygame
```

Thêm thư viện Pygame

```
pygame.init()
pygame.mixer.init()
pygame.font.init()
```

Khởi tạo pygame, pygame mixer (để chạy các tệp âm thanh) và pygame font (để tạo các phông chữ dung cho các label)

```
playing_screen = pygame.display.set_mode((window_width, window_height))
pygame.display.set_caption("Space Invaders")
```

Tạo cửa sổ chơi game với chiều cao và rộng đã khai báo trước đó, đặt tên cho cửa sổ là “space invaders”

```
name_font   = pygame.font.SysFont("verdena", 120, "bold")
```

Tạo một kiểu phông chữ với kiểu chữ Vedera cỡ chữ 120 và in đậm với “bold”

```
background_img = pygame.image.load("my_images/background.png")
background_img = pygame.transform.scale(background_img, (window_width,
window_height))
```

Load ảnh với đường dẫn vào biến và scale lại ảnh theo ý muốn

I. Các câu lệnh được sử dụng

```
background_track = pygame.mixer.Sound('my_sound_tracks/background_music.mp3')
background_track.set_volume(0.3)
```

Tải tệp âm thanh vào biến và cài âm lượng cho tệp âm thanh đó

```
background_track.play()
background_track.play()
background_track.stop()
```

chơi tệp âm thanh 1 lần, chơi tệp âm thanh lặp đi lặp lại, ngưng chơi nhạc

```
level_up = level_up_font.render("LEVEL UP!", 1, (255, 255, 102))
playing_screen.blit(level_up, (0, 0))
```

tạo ra label với phông chữ “level_up_font” chứa nội dung LEVEL UP! Có màu RGB là (255, 255, 102), sau đó vẽ label đó lên màn hình với tạo độ 0,0

```
self.mask = pygame.mask.from_surface(self.image)
```

tạo hit box cho ảnh

```
pygame.draw.rect(playing_screen, (255,69,0), (0, 0, 20, 40)
```

vẽ hình chữ nhật lên màn hình với màu RGB (255,69,0) có góc trái trên ở tọa đọ 0,0 và có chiều rộng 20, cao 40

I. Các câu lệnh được sử dụng

```
def collide(obj1, obj2):
    offset_x = obj2.x - obj1.x
    offset_y = obj2.y - obj1.y
    return obj1.mask.overlap(obj2.mask, (offset_x, offset_y)) != None
```

Hàm sẽ kiểm tra xem hitbox của 2 đối tượng có bị đè lên nhau không với câu lệnh “<mask của obj1>.overlap(<mask của obj2>, (offset_X, offset_Y))

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

Làm mới cửa sổ game, xóa tất cả hình ảnh trên cửa sổ game

```
clock = pygame.time.Clock()
```

Tạo đối tượng clock để giới hạn tốc độ khung hình

```
FPS = 60

clock.tick(FPS)
```

Giới hạn tốc độ khung hình trong vòng lặp chính của trò chơi ở 60 FPS

```
keys = pygame.key.get_pressed()
```

Trả về một mảng boolean biểu thị trạng thái các phím trên bàn phím. Mỗi phần tử trong mảng tương ứng với một phím và có giá trị `True` nếu phím đó được nhấn và `False` nếu không.

I. Các câu lệnh được sử dụng

```
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        run = False
    if event.type == pygame.KEYDOWN:
                main()
```

“pygame.event.get()” là một phương thức trong pygame.event module. Khi được gọi, nó trả về một danh sách các sự kiện đã xảy ra trong game từ lần gọi trước.
Trong ví dụ trên, vòng lặp “for event in pygame.event.get():”
sẽ lặp qua từng sự kiện trong danh sách.
“event.type” là thuộc tính trong đối tượng sự kiện, biểu thị loại của sự kiện đó.

II. Các lớp đối tượng

II. Các lớp đối tượng

Trong Python, lớp đối tượng là một cách để định nghĩa một kiểu dữ liệu mới. Nó đóng gói các thuộc tính (biến) và phương thức (hàm) liên quan lại thành một đối tượng độc lập có thể được sử dụng trong chương trình.

Lớp đối tượng chứa các đặc tính và hành vi của một đối tượng cụ thể. Đặc tính của một đối tượng được biểu diễn bằng các biến thành viên, còn hành vi của nó được biểu diễn bằng các phương thức.

Khi một lớp đối tượng được định nghĩa, nó chỉ là một mô hình hoặc mẫu. Để tạo ra một đối tượng thực tế từ lớp, chúng ta sử dụng quá trình gọi là khởi tạo. Sau đó, chúng ta có thể sử dụng đối tượng để truy cập các thuộc tính và phương thức của nó.

II. Các lớp đối tượng

Có tổng cộng 11 lớp trong chương trình. Trong đó bao gồm 7 lớp chính, các đối tượng từ lớp này sẽ được dùng để tương tác với nhau. Ngoài ra còn có 4 lớp phụ đóng vai trò trang trí hoạt ảnh cho game và không có sự tương tác với các đối tượng từ lớp khác.

Các lớp chính trong chương trình	Các lớp phụ trong chương trình
1) player 	1) portal_in 
2) minions   	2) portal_out 
3) boss 	3) explosion 
4) lazers   	4) level_up_notice
5) Item    	
6) force_shield 	
7) bunch_of_kitty 	

III. Tạo và quản lý các đối tượng

III. Tạo và quản lý các đối tượng

Sau khi tạo một đối tượng, đối tượng mới được tạo sẽ được bỏ vào một list tương ứng như ví dụ sau:

```
if level%5==0:
    new_enermy = boss()
    enermy_exist.append(new_enermy)
```

Sau khi đã có nhiều list, mỗi list chứa các đối tượng tương ứng, ta có thể bốc từng đối tượng trong list để thao tác cùng như là vẽ lên cửa sổ game hoặc kiểm tra sự va chạm giữa 2 đối tượng (sẽ được giải thích ở phần sau)

Để xóa một đối tượng khỏi game (như khi tàu địch chết hoặc đối tượng vượt khỏi màn hình) ta chỉ phải xóa nó khỏi list
Ví dụ:

```
if buff_item.y >= window_height:
    item_exist.remove(buff_item)
```

IV. Vẽ các đối tượng lên cửa sổ

IV. Vẽ các đối tượng lên cửa sổ

Để vẽ một hình ảnh lên cửa sổ pygame, ta dung lệnh: <window>.blit(<image>, (<x>, <y>))
Với window là cửa sổ cần vẽ lên, image là hình ảnh cần vẽ và x,y là tạo độ trên cửa sổ
Ví dụ:

```
playing_screen.blit(self.image, (self.x, self.y))
```

Tất cả các đối tượng trong chương trình sẽ đề có phương thức (method) draw_img() hoặc một phương thức tương tự. Các phương thức này sẽ có chức năng vẽ hình ảnh (self.image) của đối tượng lên cửa sổ game

Để vẽ các đối tượng cùng một lớp lên cửa sổ game, ta sẽ dung vòng lặp for để bốc từng đối tượng trong một list chứa các đối tượng của một class (list được tạo như đã nói ở phần III), sau đó sẽ chạy phương thức draw_image() của các đối tượng đó. Ví dụ minh họa:

```
# draw buff items
for buff_item in item_exist:
    buff_item.move_and_draw()
    if buff_item.y >= window_height:
        item_exist.remove(buff_item)
```

V. Tương tác giữa các đối tượng

V. Tương tác giữa đối tượng

Để có thể tạo ra sự tương tác giữa 2 đối tượng được tạo từ một trong 7 lớp chính trong chương trình, các đối tượng này ngoài đặc tính tọa độ self.x và self.y sẽ đều sở hữu đặc tính self.mask để thể hiện hitbox của ảnh được lưu trong đặc tính self.image của đối tượng. Như ví dụ sau:

```
class force_shield():
    def __init__(self, x=0, y=0):
        self.x = x
        self.y = y
        self.image = shield_img
        self.mask = pygame.mask.from_surface(self.image)
```

Để kiểm tra sự va chạm giữa 2 đối tượng, ta sẽ kiểm tra xem hitbox của 2 đối tượng có đè lên nhau với hàm sau:

```
def collide(obj1, obj2):
    offset_x = obj2.x - obj1.x
    offset_y = obj2.y - obj1.y
    return obj1.mask.overlap(obj2.mask, (offset_x, offset_y)) != None
```

Hàm collide sẽ yêu cầu đầu vào là 2 đối tượng cần kiểm tra sự va trạm, 2 đối tượng đầu vào sẽ đều phải sở hữu đặc tính self.x, self.y và self.mask. Hàm sẽ trả về toán tử Boolean tương ứng True khi hitbox giữa 2 đối tượng đè (overlap) lên nhau và False khi không thỏa điều kiện

VI. Cấu trúc vòng lặp chính

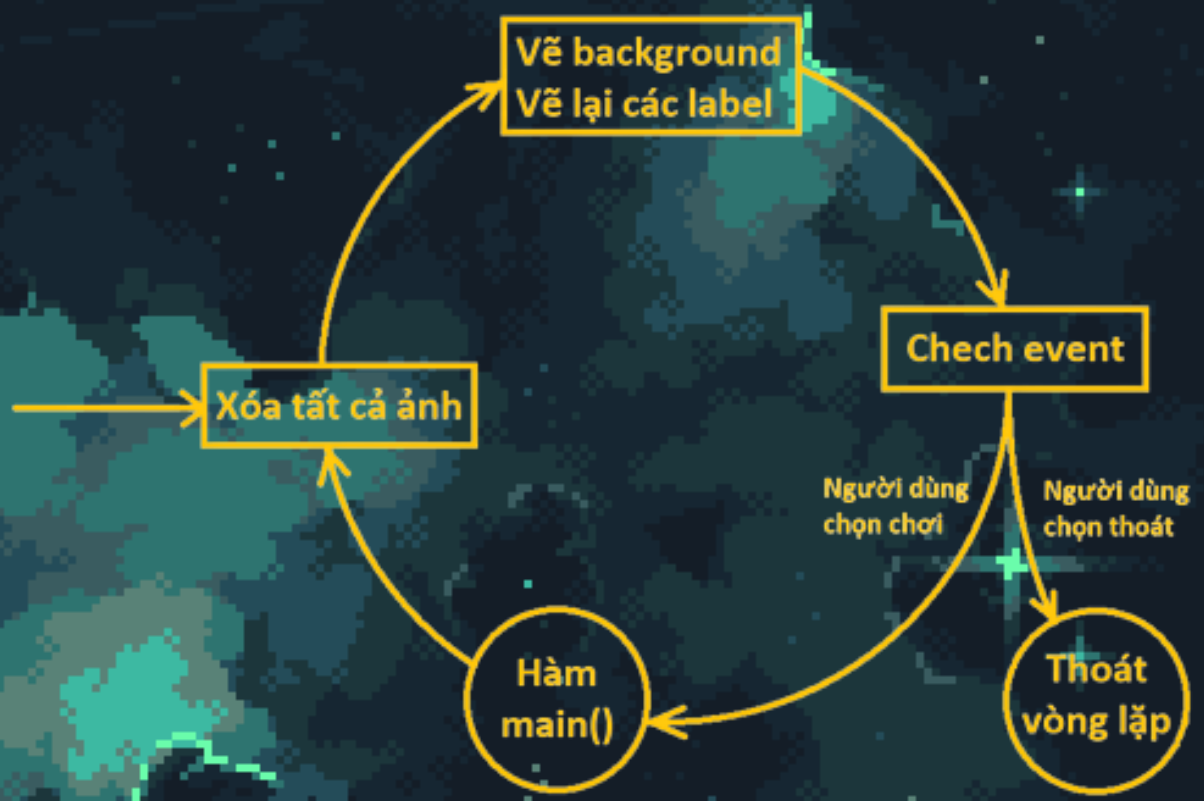
VI. Cấu trúc vòng lặp

a. Vòng lặp màn hình menu

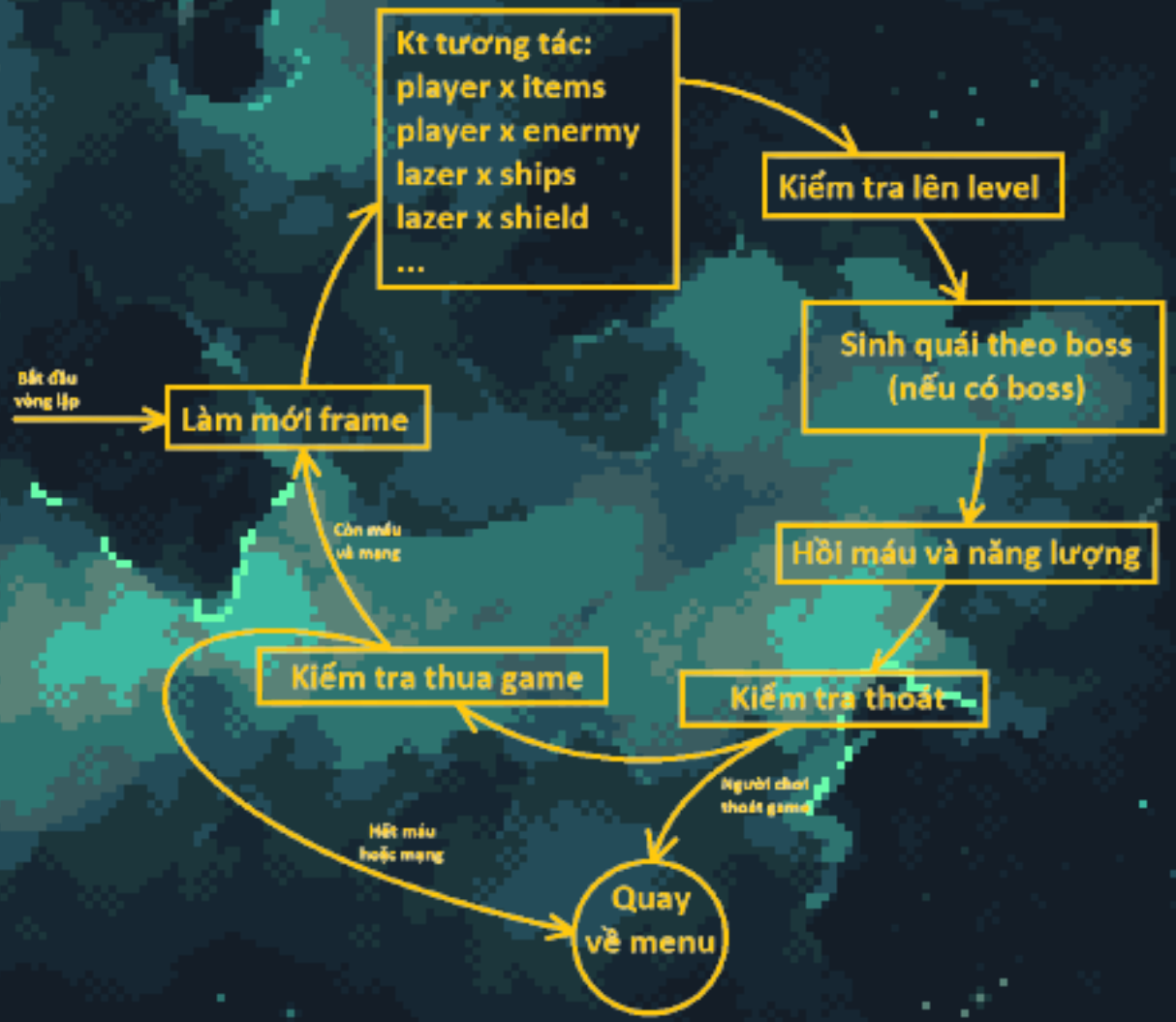
Sau khi chạy chương trình, người dung sẽ được đưa vào màn hình menu với vòng lặp trong hàm lobby()

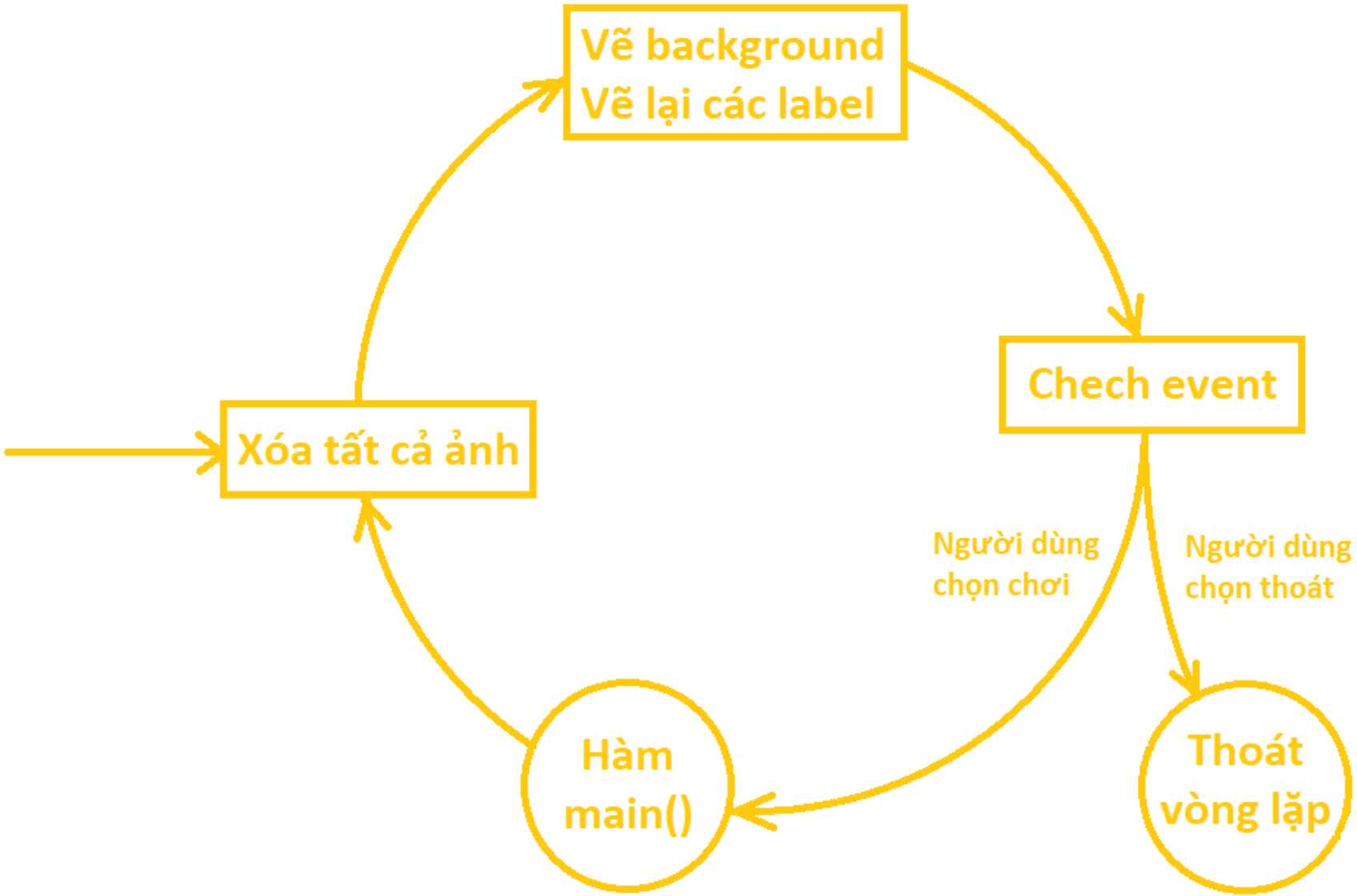
vòng lặp này sẽ liên tục làm mới cửa sổ game và kiểm tra sự kiện từ người chơi. Nếu người chơi nhấn nút thoát thì sẽ thoát khỏi vòng lặp và kết thúc chương trình. Nếu người chơi bấm bất kì nút nào khác thì sẽ được đưa vào vòng lặp mới trong hàm main() nơi trò chơi được khởi động.

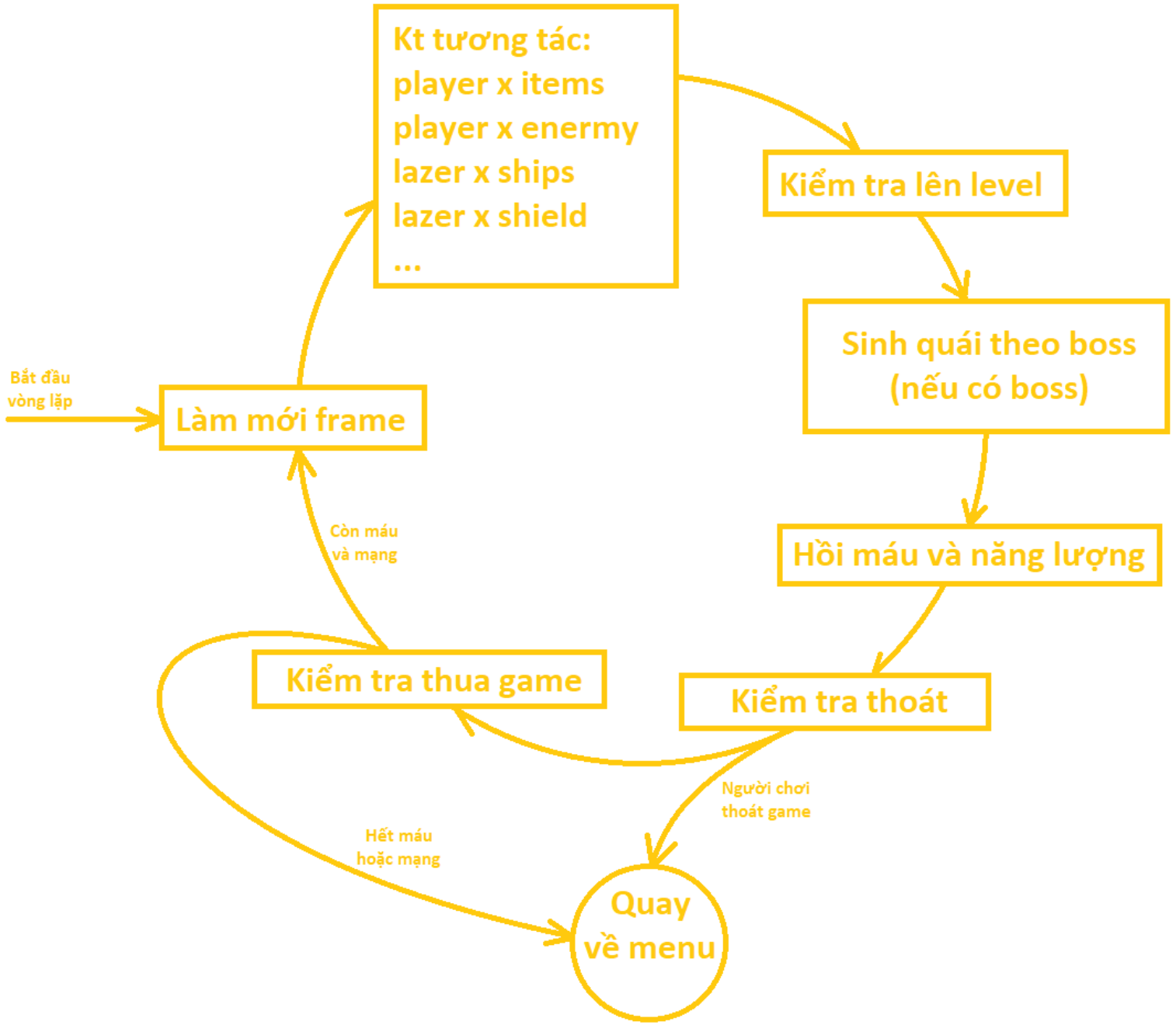
Nếu người chơi đã thua trong game, thì sẽ thoát khỏi vòng lặp trong hàm main() và quay lại vòng lặp trong hàm lobby() và đợi sự kiện tiếp theo của người chươi



VI. Cấu trúc vòng lặp
b. Vòng lặp trong game




Vẽ background
Vẽ lại các label
Chech event
Người dùng
chọn chơi
Người dùng
chọn thoát
Xóa tất cả ảnh
Hàm
main()
Thoát
vòng lặp


Kt tương tác:
player x items
player x enermy
lazer x ships
lazer x shield
...
Kiểm tra lên level
Sinh quái theo boss
(nếu có boss)
Hồi máu và năng lượng
Kiểm tra thoát
Người chơi
thoát game
Kiểm tra thua game
Còn máu
và mạng
Làm mới frame
Bắt đầu
vòng lặp
Hết máu
hoặc mạng
Quay
về menu