TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

TÊN ĐỀ TÀI

LẬP TRÌNH PYTHON

Họ tên thành viên trong nhóm

DƯƠNG THÀNH TRƯỞNG

LÊ ĐÌNH MẢNH

TRƯƠNG GIA MINH

GIÁO VIÊN HƯỚNG DẪN:

TS. TRỊNH TẤN ĐẠT

THS. NGUYỄN TRUNG TÍN

I. Lời mở đầu:

Tên đề tài: DINOSAUR and friends

Mục đích nghiên cứu:

1. Nghiên cứu và áp dụng các kỹ thuật lập trình và thiết kế game để tạo ra một trò chơi chất lượng cao.

2. Phát triển kỹ năng lập trình và thực hành các phương pháp lập trình game thông qua việc sử dụng Pygame.

3. Nghiên cứu và áp dụng các công nghệ mới trong lĩnh vực game để cải thiện hiệu suất và chất lượng của game.

4. Tạo ra một trò chơi hấp dẫn và thú vị để thử nghiệm các kỹ thuật lập trình, thiết kế và nghiên cứu game.

5. Nghiên cứu và phát triển các tính năng mới cho trò chơi như đồ họa, âm thanh, gameplay, tính năng mạng, v.v.

6. Tạo ra một sản phẩm game hoàn chỉnh và thực tế có thể được phát hành trên nhiều nền tảng và được người dùng yêu thích.

7. Nghiên cứu và thực hành quy trình phát triển game chuyên nghiệp để có thể tham gia vào ngành công nghiệp game chuyên nghiệp.

II. Mục lục

Contents

III. Nội dung

1. *Giới thiệu về đề tài thực hiện và các thành viên trong nhóm*

Đề tài thực hiện: lập trình game python sử dụng pygame

Tên đề tài: DINOSAUR AND FRIENDS

Các thành viên trong nhóm:

Họ và tên	mã số sinh viên
- Lê Đình Mảnh	3121410315
- Trương Gia Minh	3121410324
- Dương Thành Trưởng	3121410546

Ngày hoàn thành

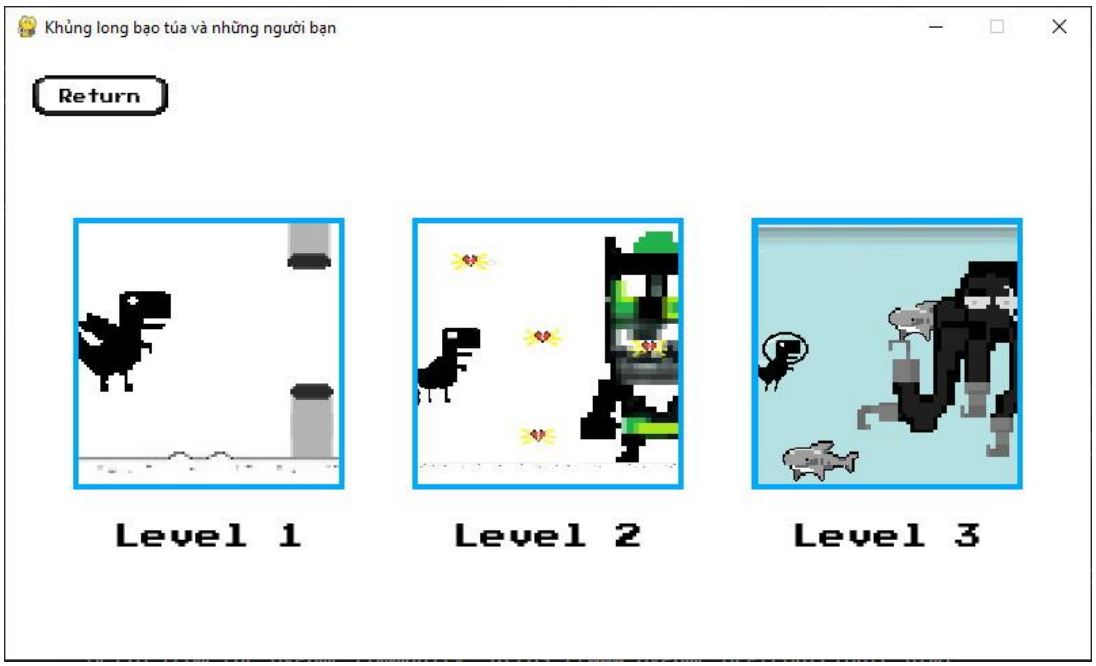
2. *Giới thiệu về game DINOSAUR AND FRIENDS*

- **Ý tưởng**

  Dựa trên game flappy bird và game dinosaurous.

- **Cách chơi**

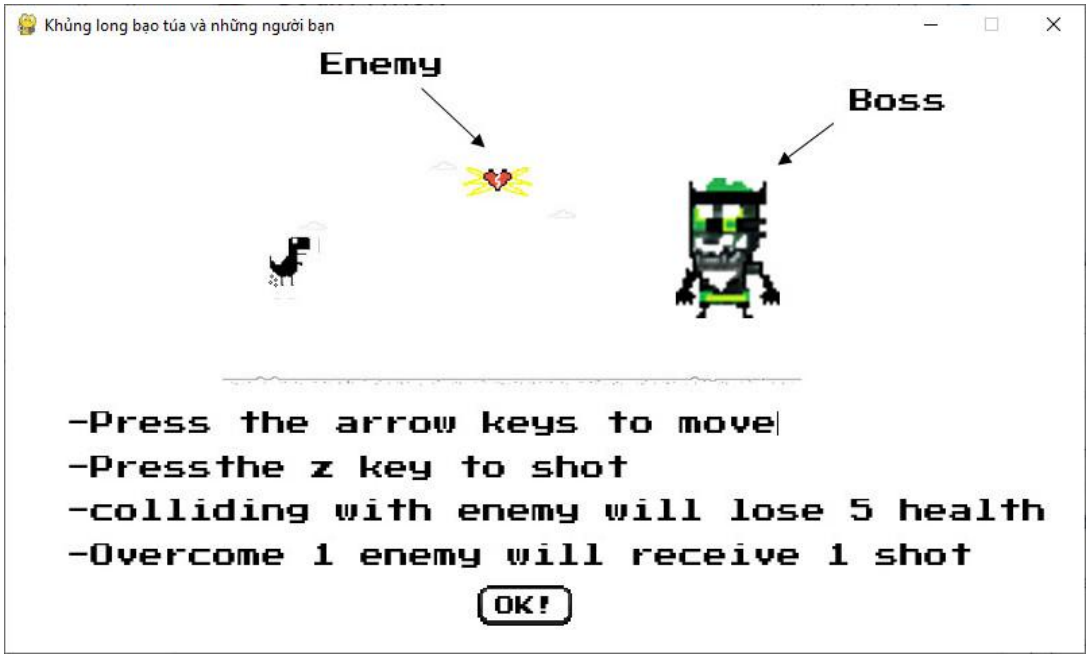
  Game gồm 3 màn:



Màn 1:

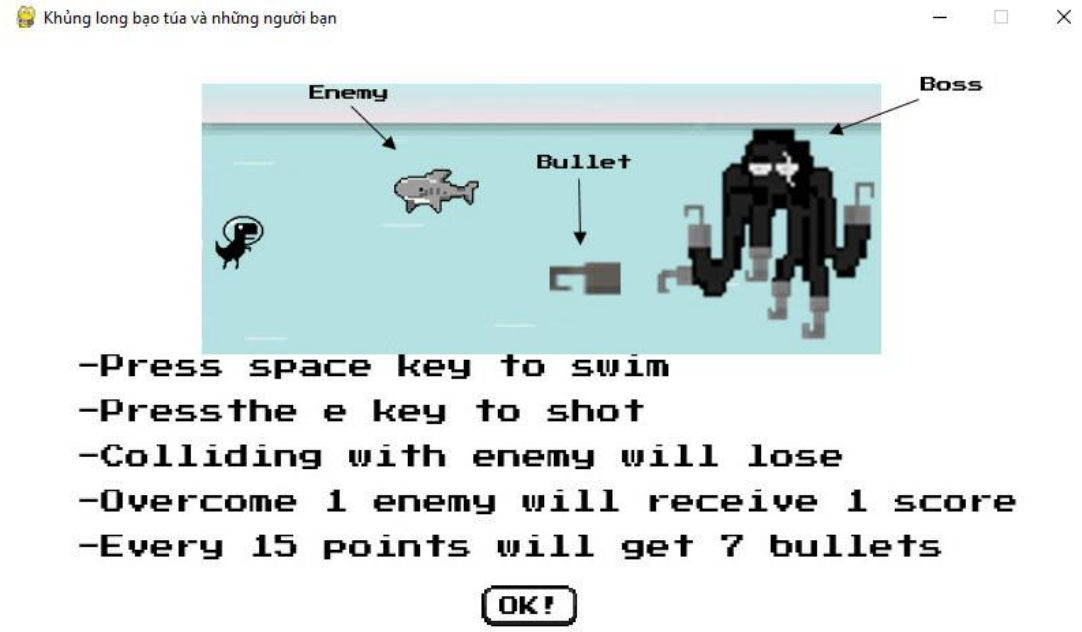
Sử dụng phím mũi tên lên hoặc phím space để cho khủng long bay lên, vượt qua khe giữa 2 ống xuất hiện trên và dưới trong quá trình di chuyển của khủng long, nếu khủng long chạm ống, chạm sàn hoặc lên quá cao so với khung ảnh thì trò chơi kết thúc. Khi đạt điểm 100 chúng ta được phép mở khóa màn tiếp theo nếu ở màn 1 này khủng long chạm ống, bị rơi xuống sàn hoặc lên quá cao trong khung ảnh thì chúng ta có 2 lựa chọn là chơi lại hoặc chơi màn tiếp theo.

Màn 2:



Sử dụng 4 phím mũi tên để điều khiển khủng long tránh vật thể được boss của màn thả ra, khi vật thể vượt qua cạnh trái của khung ảnh thì sẽ tích được 1 điểm để phản công boss, sử dụng phím z để tấn công. Khi hạ được boss sẽ mở khóa màn tiếp theo.

Màn 3:



Sử dụng phím space giúp khủng nổi lên mặt nước mỗi lần vượt qua 15 con cá mập sẽ tích được 7 viên đạn để phản công, đạt 17 điểm boss sẽ xuất hiện và tấn công

khủng long, để phản công chúng ta nhấn phím e. khi vượt qua màn này (màn cuối) chúng ta sẽ đến với video kết thúc.

- **Tính năng của trò chơi**

1. lưu màn đang chơi

```python
def save_data(localData = 0, Data1 = 0, Data2 = "t"):
    f = open("data.txt", mode='r')
    r = f.readline().split(",")
    r[localData] = str(Data1)
    r[localData + 3] = Data2
    s = r[0] + "," + r[1] + "," + r[2] + "," + r[3] + "," + r[4] + "," + r[5]
    f.close()
    f = open("data.txt", mode='w')
    f.write(s)
    f.close()
```

2. chơi lại nhiều lần

```python
def announcement(checkWin, Score, highScore, Screen):
        Background = None
        if checkWin:
            Background = pygame.transform.scale(pygame.image.load(f"imgdino/startgame/background1.png"), (200, 200))
            Sounds["winGame"].play()
        else:
            Sounds["loseGame"].play()
            Background = pygame.transform.scale(pygame.image.load(f"imgdino/startgame/background2.png"), (200, 200))
        Clock = pygame.time.Clock()
        Replay = Button(50, 200, 100, 50, "Replay", 15)
        Exit = Button(650, 200, 100, 50, "Exit", 15)
        Next = Button(305, 370, 190, 50, "Next level", 15)
        scoreText = pygame.font.Font(typeText, 20).render(f"Scrore: {int Score }", False, "black")
        highScoreText = pygame.font.Font(typeText, 20).render(f"High scrore: {int highScore}", False, "yellow")
        backgroundRect = pygame.Rect(295, 25, 210, 400)
        while(True):
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    sys.exit()
            if backgroundRect.width < 750:
                backgroundRect.x -= 10
                backgroundRect.width += 20
                pygame.draw.rect(Screen, "gray", backgroundRect, 0, 50)
                pygame.display.update()
                Clock.tick(60)
                continue
```

```python
                pygame.draw.rect(Screen, "gray", (25, 25, 750, 400), 0, 50)
                if Replay.buttonEvent(Screen):
                    return True, True
                if Exit.buttonEvent(Screen):
                    return False, False
                if checkWin:
                    if Next.buttonEvent(Screen):
                        return False, True
                Screen.blit(scoreText, scoreText.get_rect(center = (150, 70)))
                Screen.blit(highScoreText, highScoreText.get_rect(center = (600, 70)))
                Screen.blit(Background, Background.get_rect(center = (400, 250)))
                pygame.display.update()
                Clock.tick(10)
```

3. lưu dữ liệu trò chơi khi tắt ứng dụng

4. có nhạc nền (có thể bật tắt)

```
8   class Main():
9       #Khởi tạo game
10      def __init__(self, Width, Height):
11          pygame.init()
12          pygame.display.set_caption('Khủng long bạo túa và những người bạn')
13          Sounds["musicgame"] = pygame.mixer.Sound("imgdino/startgame/musicstartgame.wav")
14          Sounds["Click"] = pygame.mixer.Sound('imgdino/startgame/tap.wav')
15          Sounds["winGame"] = pygame.mixer.Sound('imgdino/startgame/wingame1.wav')
16          Sounds["loseGame"] = pygame.mixer.Sound('imgdino/startgame/losegame1.wav')
17          Sounds["musicgame"].set_volume(0.1)
18          Sounds["musicgame"].play(loops = -1)
```

5. Bật tắt nhạc nền:

```
def set_volume(value):
    for Sound in Sounds:
            if Sound == "winGame" or Sound == "loseGame":
                Sounds[Sound].set_volume(value/2)
                continue
            Sounds[Sound].set_volume(value)

def get_volume():
    if Sounds["Click"].get_volume() == 0:
        return False
    return True
```

6. có âm thanh khi thao tác

```
24      def buttonEvent(self, Screen):
25          mousePos = pygame.mouse.get_pos()
26          if self.Rect.collidepoint(mousePos):
27              if self.Text == "":
28                  Screen.blit(self.backGround[0], (self.X, self.Y))
29              else:
30                  Screen.blit(self.backGround[1], (self.X, self.Y))
31              if pygame.mouse.get_pressed(num_buttons = 3)[0]:
32                  Sounds["Click"].play()
```

Âm thanh Màn 1:

```
32          #tạo âm thanh
33          Sounds["dinoSoundMap1"] = pygame.mixer.Sound('imgdino/map1/sfx_wing.wav')
34          Sounds["hitSoundMap1"] = pygame.mixer.Sound('imgdino/map1/sfx_hit.wav')
35          Sounds["scoreSoundMap1"] = pygame.mixer.Sound('imgdino/map1/te.wav')
36
```

Âm thanh Màn 2:

```
14      Sounds["dinoShoot"] = pygame.mixer.Sound('imgdino/startgame/shoot1.wav')
15      Sounds["bossCollide"] = pygame.mixer.Sound('imgdino/startgame/tick.wav')
```

Âm thanh Màn 3:

```
Sounds["dinoShoot2"] = pygame.mixer.Sound('imgdino/startgame/shoot1.wav')
Sounds["bossHit"] = pygame.mixer.Sound("imgdino/startgame/hitboss.wav")
Sounds["ItemSound"] = pygame.mixer.Sound("imgdino/startgame/Item.wav")
```

3. *Công nghệ và công cụ được sử dụng*

- **Công nghệ**

  + ngôn ngữ lập trình python

  + thư viện pygame

- **Công cụ**

  + vẽ pixel art

  + vs code

  + notepad, google, youtube

4. *Quy trình phát triển*

- **Lựa chọn đề tài**

  1. lý do lựa chọn đề tài:

     Có niềm đam mê lớn với lập trình game đặc biệt là game 2D

     Thử sức và phát triển kỹ năng

     Phát triển sản phẩm phục vụ mục đích giải trí

     Tiềm năng kinh doanh, thương mại

     Thử nghiệm và tìm hiểu công nghệ mới

- **Phân tích đồ án**

  1. các bước phân tích đồ án:

     Xác định yêu cầu đồ án

     Lên ý tưởng

     Thiết kế

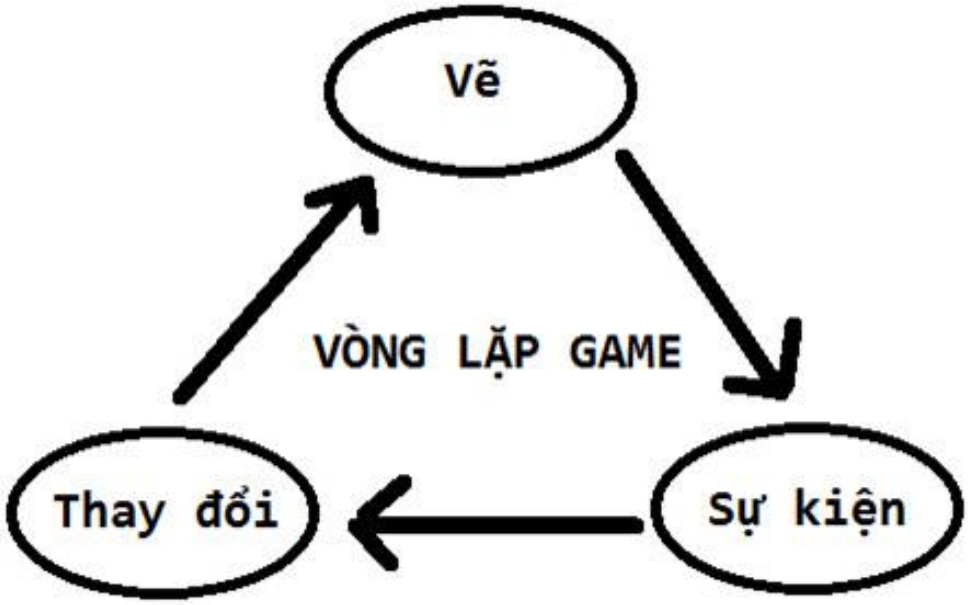
- **Phân chia công việc**
- **Tìm kiếm thông tin**

- **Lọc thông tin cần thiết**
- **Thực hiện công việc**
- **Kiểm thử chương trình**
    1. Kiểm tra tính đúng đắn của chương trình
    2. Kiểm tra tính ổn định
    3. Kiểm tra tính bảo mật
    4. Kiểm tra hiệu suất
    5. Kiểm tra tính năng
    6. Kiểm tra giao diện
- **Ghép code**
- **Điều chỉnh chương trình**
    1. Phân tích và xác định vấn đề
    2. Sửa lỗi
- **Chạy tổng thể chương trình**
- **Phát triển thêm**
    1. Thêm chế độ chơi mới
    2. Thêm bản đồ mới
    3. Thêm nhân vật mới
    4. Thêm tính năng mới
    5. Thêm chức năng kết nối với các trò chơi khác
    6. Thêm hệ thống nhiệm vụ
- **Tối ưu và cải tiến**
- **Hoàn thành sản phẩm**

5. ***Kiến trúc và thiết kế***

- **Kiến trúc**

  game có 3 việc chính tạo thành vòng lập: Vẽ, bắt sự kiện, thay đổi đối tượng.



- **Thiết kế**
  1. Thiết kế hướng đối tượng

     Cho phép tập trung vào các đối tượng riêng lẻ trong trò chơi, cho phép các đối tượng tương với nhau và quản lý chúng dễ dàng hơn.
  2. Thiết kế mô hình - chế độ xem – điều khiển

     Đây là một thiết kế phổ biến cho các trò chơi 2D và 3D. Nó bao gồm ba phần chính là mô hình (model), chế độ xem (view) và điều khiển (controller). Mô hình đại diện cho dữ liệu trong trò chơi, chế độ xem đại diện cho cách dữ liệu được hiển thị trên màn hình, và điều khiển đại diện cho cách người dùng tương tác với trò chơi.
  3. Thiết kế hệ thống sự kiện:

     Đây là một phương pháp thiết kế cho các trò chơi có nhiều tương tác từ người chơi. Hệ thống sự kiện cho phép lập trình viên đăng ký các sự kiện từ người chơi, ví dụ như nhấn nút chuột, nhấn phím, và tương tác trên màn hình. Khi sự kiện xảy ra, hệ thống sẽ gọi các phương thức tương ứng để xử lý sự kiện đó.
- **Các thành phần trong game.**
  1. Game engine: Đây là thành phần quan trọng nhất của một chương trình game, chịu trách nhiệm điều khiển và quản lý toàn bộ game. Game engine cũng chịu trách nhiệm tính toán và xử lý các tương tác của người chơi với game.

2. Đồ họa (Graphics): Là thành phần đảm nhiệm trình diễn hình ảnh của game, đồ họa giúp tạo ra các đối tượng, nhân vật, cảnh quan và các hiệu ứng đặc biệt trong game.

3. Âm thanh (Sound): Cung cấp các âm thanh và nhạc nền cho game. Âm thanh giúp tăng tính tương tác của game với người chơi.

4. Điều khiển và điều hướng (Control and Navigation): Điều khiển và điều hướng giúp người chơi điều khiển các nhân vật trong game hoặc các thành phần khác trong game.

5. Cơ chế điều khiển (Game mechanics): Đây là thành phần quyết định cách thức và quy tắc của game. Cơ chế điều khiển quy định các thao tác của người chơi và cách thức tính điểm trong game.

6. Cấu trúc dữ liệu (Data structures): Cung cấp cấu trúc dữ liệu cho game. Các cấu trúc này bao gồm các thông tin về các đối tượng, các nhân vật và các đối tượng khác trong game.

7. Lưu trữ dữ liệu (Data storage): Lưu trữ dữ liệu giúp lưu lại các thông tin về các trò chơi đã chơi, điểm số và các thông tin khác.

- **Cách chúng tương tác với nhau**
  1. Các thành phần này tương tác với nhau thông qua các phương thức giao tiếp, bao gồm cả giao tiếp giữa các đối tượng trong game và giao tiếp với người chơi thông qua bộ điều khiển. Việc thiết kế các phương thức giao tiếp này sẽ quyết định độ mượt mà và tính tương tác của trò chơi với người chơi.
- **Một số yếu tố khác tác động tới kiến trúc và thiết kế của game**
  1. Đồ họa và âm thanh: Đồ họa và âm thanh chất lượng cao sẽ tạo ra trải nghiệm chơi game tốt hơn.

2. Kiểm soát thời gian: Game cần kiểm soát thời gian sao cho người chơi không quá nản lòng hoặc nhàm chán.

6. ***Phân tích dự án***

- **Mô tả giao diện game**

1. đầu tiên chúng ta sẽ đến với giao diện chính của game:

Bao gồm các chức năng:

Start game dùng để bắt đầu game với màn đầu tiên

Phím load save dùng để chọn màn với điều kiện màn đó đã được mở khóa trong quá trình chơi trước đó của người chơi

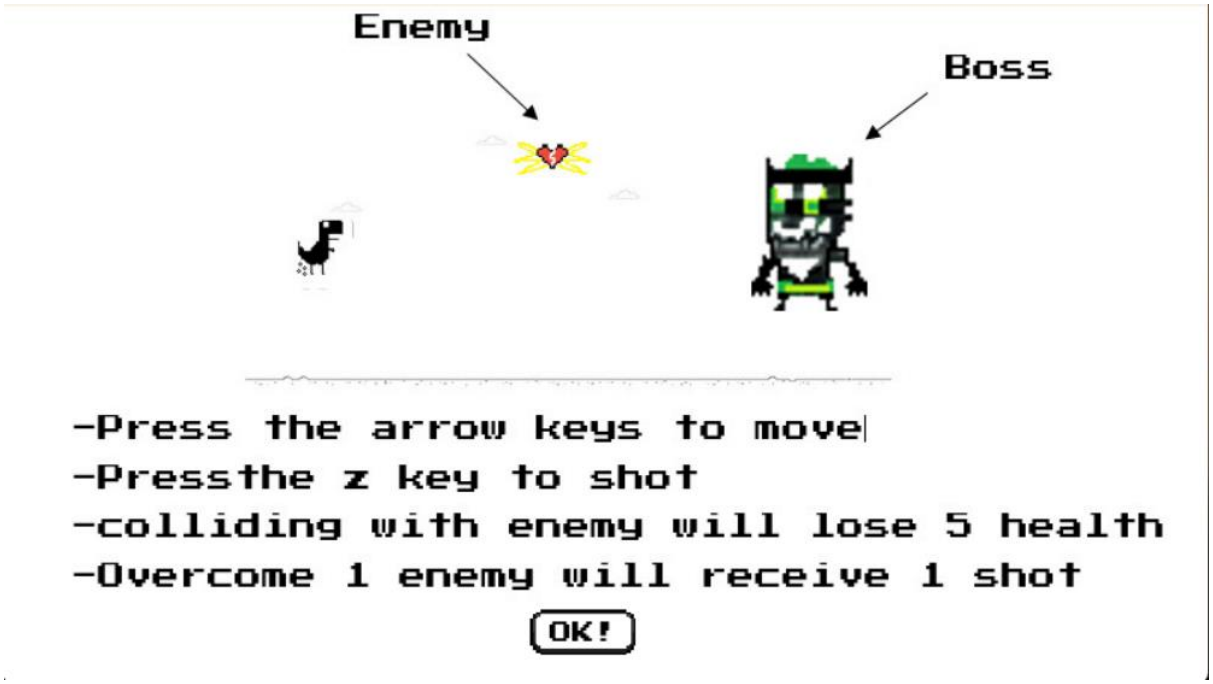
Menu để điều chỉnh âm lượng trong game

2. Khi chọn start game, chúng ta sẽ đến với màn hình hướng dẫn của màn chơi đầu tiên

Nhấn phím ok để bắt đầu trò chơi, ta thấy giao diện của màn chơi đầu tiên

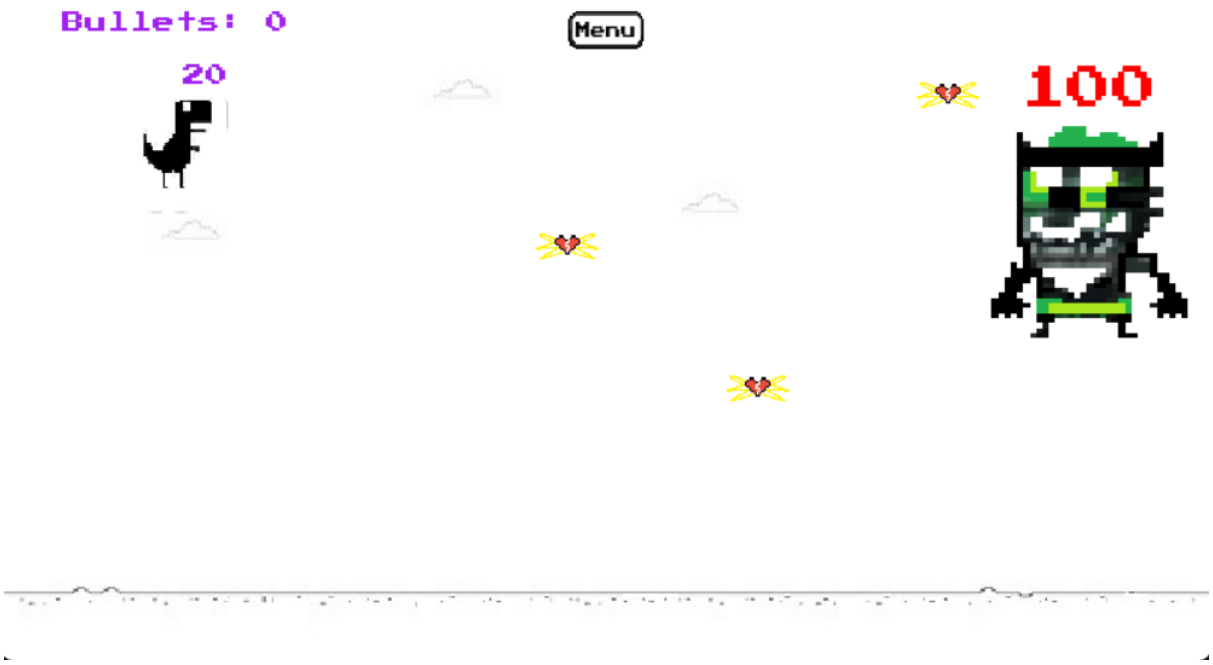
điểm sẽ tăng dần theo thời gian với điều kiện người chơi ko chạm vào chướng ngại vật

3. Khi đạt 100 điểm ở màn chơi thứ nhất chúng ta được phép lựa chọn chuyển sang màn tiếp theo



Nhấn vào ok để tham gia màn này

di chuyển khủng long để né vật thể từ boss, sau đó tích điểm để phản công lại boss

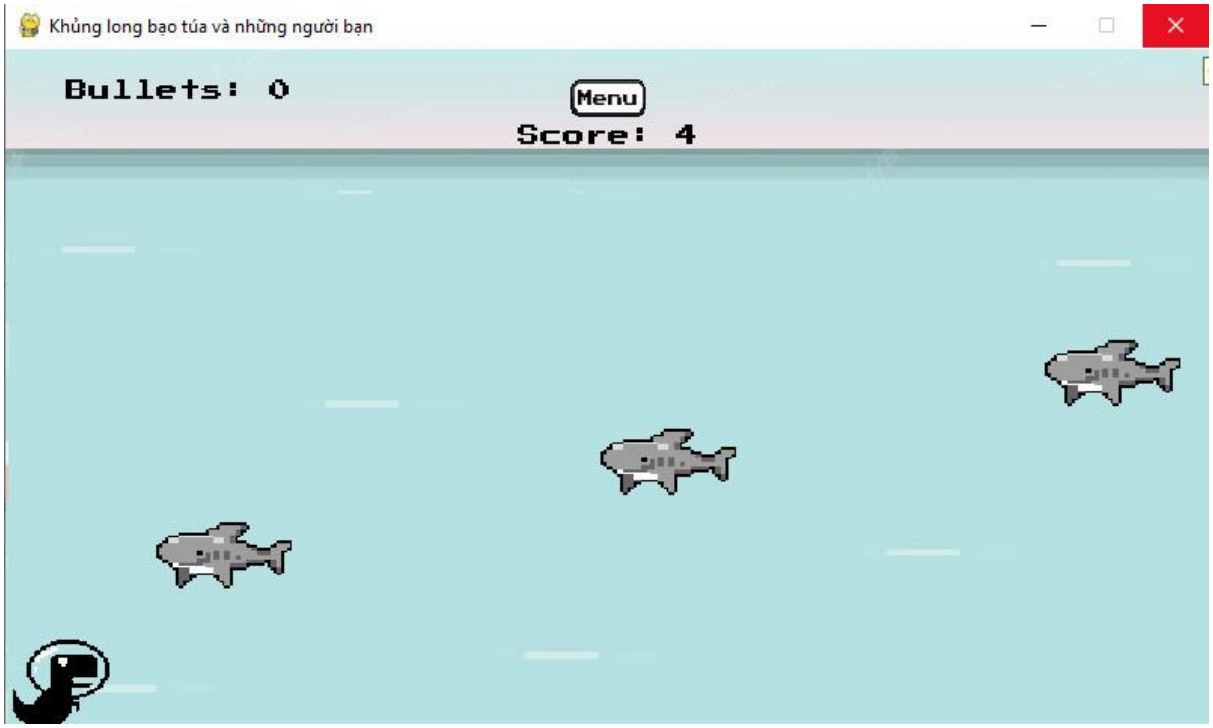


trong quá trình chơi phải giữ điểm sinh mệnh, không được để điểm sinh mệnh về 0. Nếu không người chơi sẽ thất bại ở màn chơi này.

cố gắng tránh những vật thể cho đến khi có vật thể đi qua cạnh trái màn hình sẽ tích được đạn để phản công.

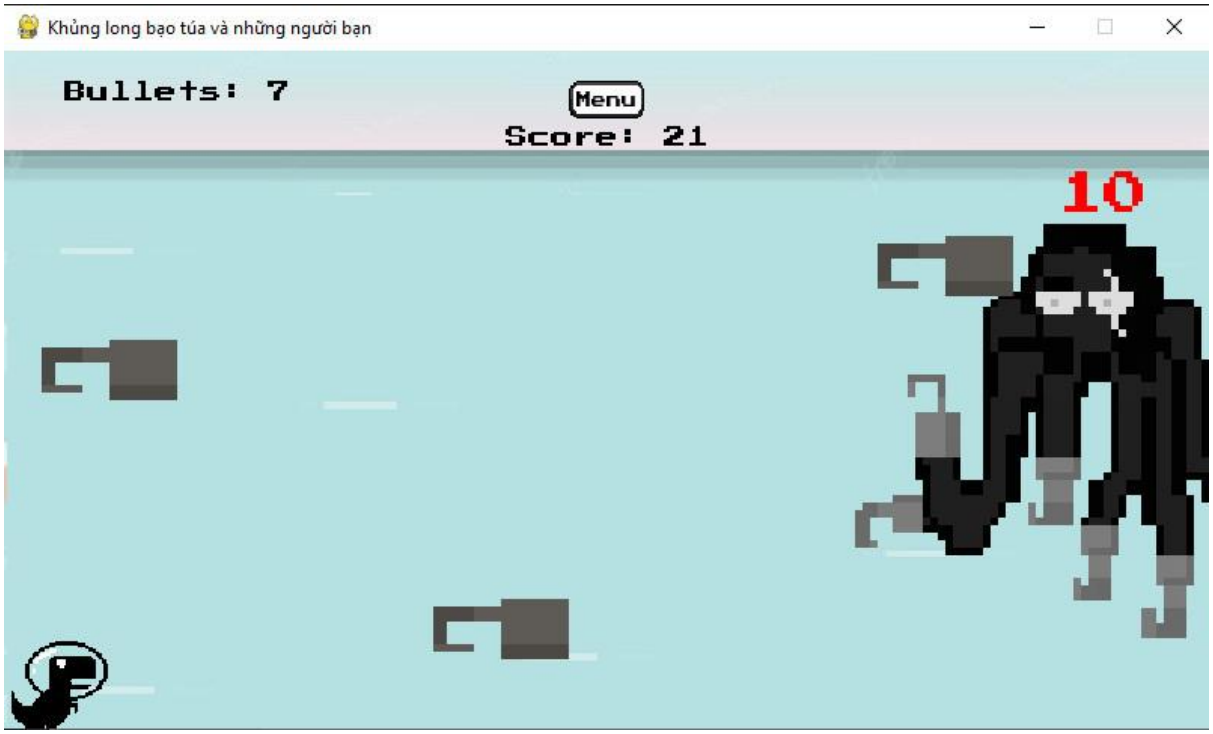
4. hạ được boss để qua màn tiếp theo.

Nhấn ok để vào màn chơi



đối với màn chơi cuối cùng, người chơi sẽ kết hợp cách chơi giữa 2 màn trước để giành chiến thắng chung cuộc.

khi điểm số đạt 17 boss cuối sẽ xuất hiện



hạ boss để giành chiến thắng, và giải cứu tình nhân của mình.

nếu tổng điểm của người chơi không vượt qua 200 điểm thì badending sẽ diễn ra =))

5. Nếu người chơi vượt qua mỗi màn chơi sẽ xuất hiện giao diện game win

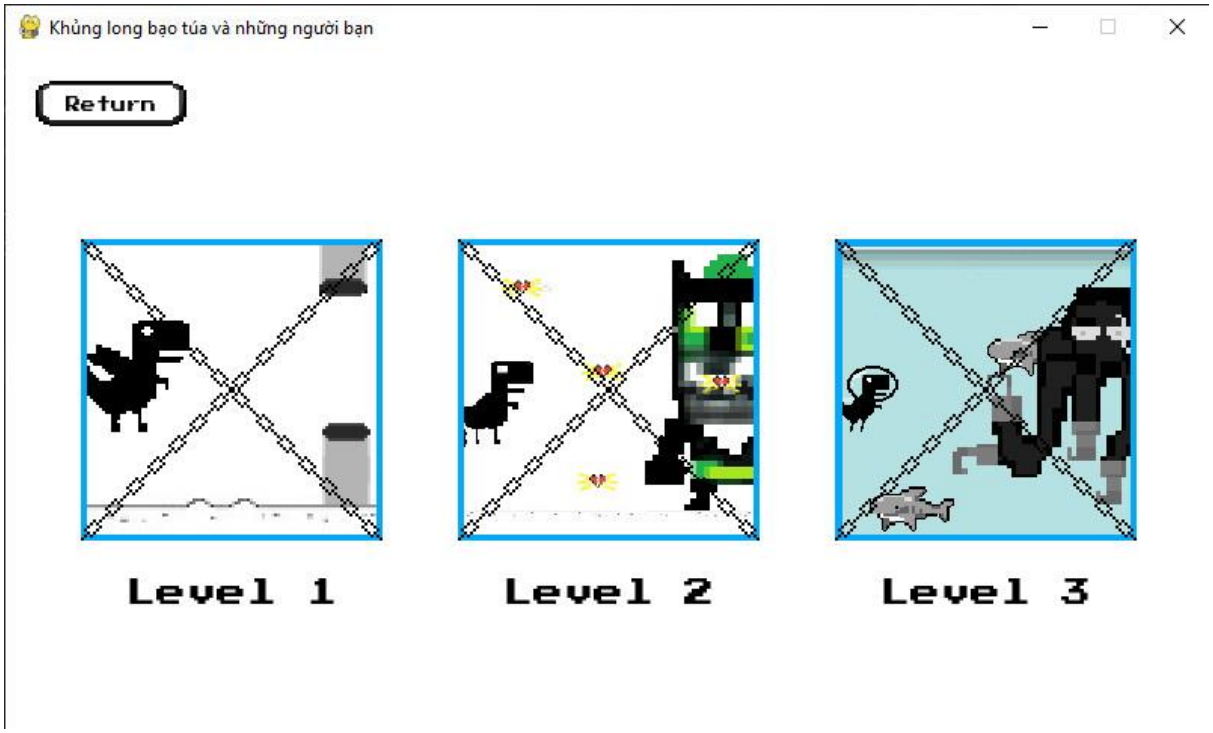
6. Ngược lại nếu người chơi không qua được chạm vào chướng ngại vật thì sẽ hiển thị giao diện game lose

7. Khi chọn Load save

Sẽ xảy ra 2 trường hợp

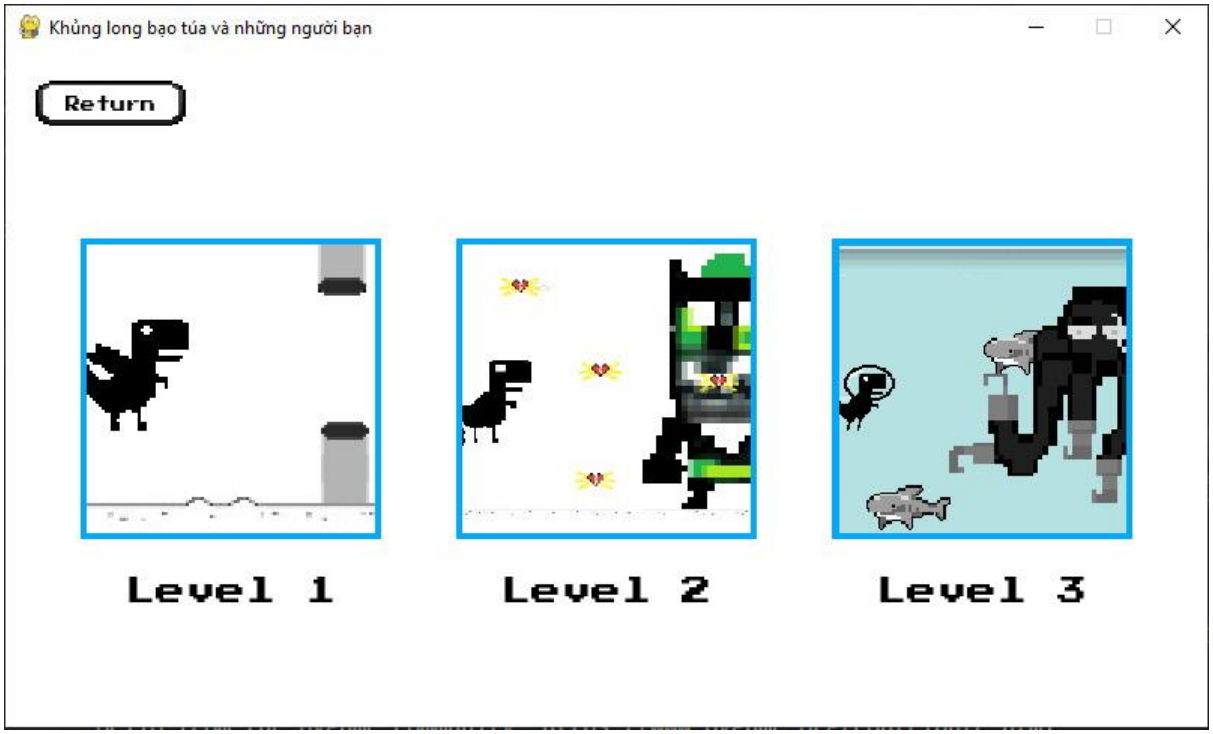
Nếu người chơi chưa qua được bất kì màn chơi nào thì giao diện hiện ra như bên dưới.

Người chơi không được phép chọn màn trong giao diện này



Nếu người chơi đã qua được tất cả các màn chơi

Người chơi được phép chọn màn trong giao diện này



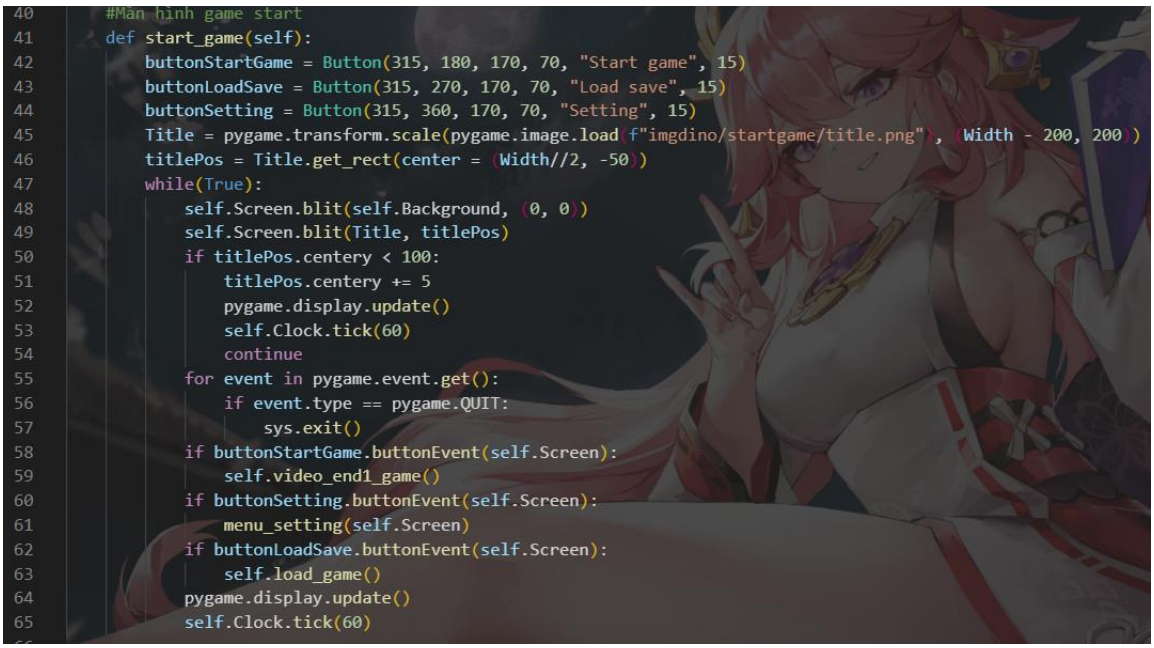
8. Khi chọn setting:

   Mặc định âm thanh sẽ mở

Chọn vào biểu tượng chiếc loa để tắt âm thanh

- **Nguyên lý hoạt động:**

Game:



```
#Màn hình game start
def start_game(self):
    buttonStartGame = Button(315, 180, 170, 70, "Start game", 15)
    buttonLoadSave = Button(315, 270, 170, 70, "Load save", 15)
    buttonSetting = Button(315, 360, 170, 70, "Setting", 15)
    Title = pygame.transform.scale(pygame.image.load(f"imgdino/startgame/title.png"), (Width - 200, 200))
    titlePos = Title.get_rect(center = (Width//2, -50))
    while(True):
        self.Screen.blit(self.Background, (0, 0))
        self.Screen.blit(Title, titlePos)
        if titlePos.centery < 100:
            titlePos.centery += 5
            pygame.display.update()
            self.Clock.tick(60)
            continue
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit()
        if buttonStartGame.buttonEvent(self.Screen):
            self.video_end1_game()
        if buttonSetting.buttonEvent(self.Screen):
            menu_setting(self.Screen)
        if buttonLoadSave.buttonEvent(self.Screen):
            self.load_game()
        pygame.display.update()
        self.Clock.tick(60)
```

Gồm các nút bấm start game, load game, menu

Phím start game dùng để vào màn chơi đầu tiên của game

Phím load game dùng để chọn màn với điều kiện màn đó đã được mở khóa trong quá trình chơi trước đó của người chơi

Menu để điều chỉnh âm lượng trong game

Màn 1:

```python
class map1():
    def __init__(self, Surface):
        super().__init__()
        #thiết lập hình ảnh
        self.Background = pygame.transform.scale(pygame.image.load(f"imgdino\\map1\\background0.png").convert_alpha(), (Width, Height))
        self.floorXPos = 0
        self.Floor = pygame.transform.scale2x(pygame.image.load("imgdino/map1/background3.png").convert())
        self.dinoDown = pygame.image.load('imgdino/map1/dino-fly-down.png')
        self.dinoMid = pygame.image.load('imgdino/map1/dino-fly.png')
        self.dinoUp = pygame.image.load('imgdino/map1/dino-fly-up.png')
        self.dinoList= [self.dinoDown, self.dinoMid, self.dinoUp]
        self.dinoIndex = 0
        self.Dinosaur = self.dinoList[self.dinoIndex]
        self.dinoRect = self.Dinosaur.get_rect(center = (200, 100))
        #Thiết lập trọng lực
        self.Gravity = 0.15
        self.dinoMovement = 0
        #Thiết lập thông tin màn chơi
        self.gameActive = True
        self.Score = 0
        self.highScore = 0
        self.Block = True
        self.gameFont = pygame.font.Font(typeText, 20)
        #tạo ống
        self.pipeSurface = pygame.transform.scale2x(pygame.image.load("imgdino/map1/pipe.png"))
        self.pipeList = []
        self.pipeHeightList =[200, 250, 300]
        #tạo âm thanh
        Sounds["dinoSoundMap1"] = pygame.mixer.Sound('imgdino/map1/sfx_wing.wav')
        Sounds["hitSoundMap1"] = pygame.mixer.Sound('imgdino/map1/sfx_hit.wav')
        Sounds["scoreSoundMap1"] = pygame.mixer.Sound('imgdino/map1/te.wav')
        #tạo khung hình
        self.Surface = Surface
        self.Clock = pygame.time.Clock()
```

Sử dụng trọng lực và tốc độ di chuyển khung ảnh để cho khủng long chuyển động,

```python
self.dinoMovement += self.Gravity
rotated_dino = self.rotate_dino()
self.dinoRect.centery += self.dinoMovement
self.Surface.blit(rotated_dino,self.dinoRect)
```

```python
def rotate_dino(self):
    new_dino= pygame.transform.rotozoom(self.Dinosaur,-self.dinoMovement*3,1)
    return new_dino
```

các ống (vật cản) sẽ xuất hiện trên đường đi một cách ngẫu nhiên về chiều dài lẫn tọa độ của khung hình,

```python
def create_pipe(self):
    randomPipePos =random.choice(self.pipeHeightList)
    bottomPipe = self.pipeSurface.get_rect(midtop = (800, randomPipePos))
    topPipe = self.pipeSurface.get_rect(midtop = (800, randomPipePos - 650))
    return bottomPipe, topPipe
```

số điểm sẽ tăng lên theo thời gian với điều kiện khủng long không chạm vào bất kì vật cản nào trên đường đi, chạm sàn và chạm nóc khung hình.

```
            self.Score += 0.1
            self.score_display('main game')
            if self.Score % 100 == 0 and self.Score != 0:
                Sounds["scoreSoundMap1"].play()
        elif self.Score > 100:
            self.update_score()
```

```
def score_display(self, gameState):
    if gameState=='main game':
        scoreSurface = self.gameFont.render(f'Score: {int(self.Score)}',True, 0,0,0)
        scoreRect= scoreSurface.get_rect(topleft = (10,30))
        self.Surface.blit(scoreSurface,scoreRect)

def update_score(self):
    if self.Score > self.highScore:
        self.highScore = self.Score
```

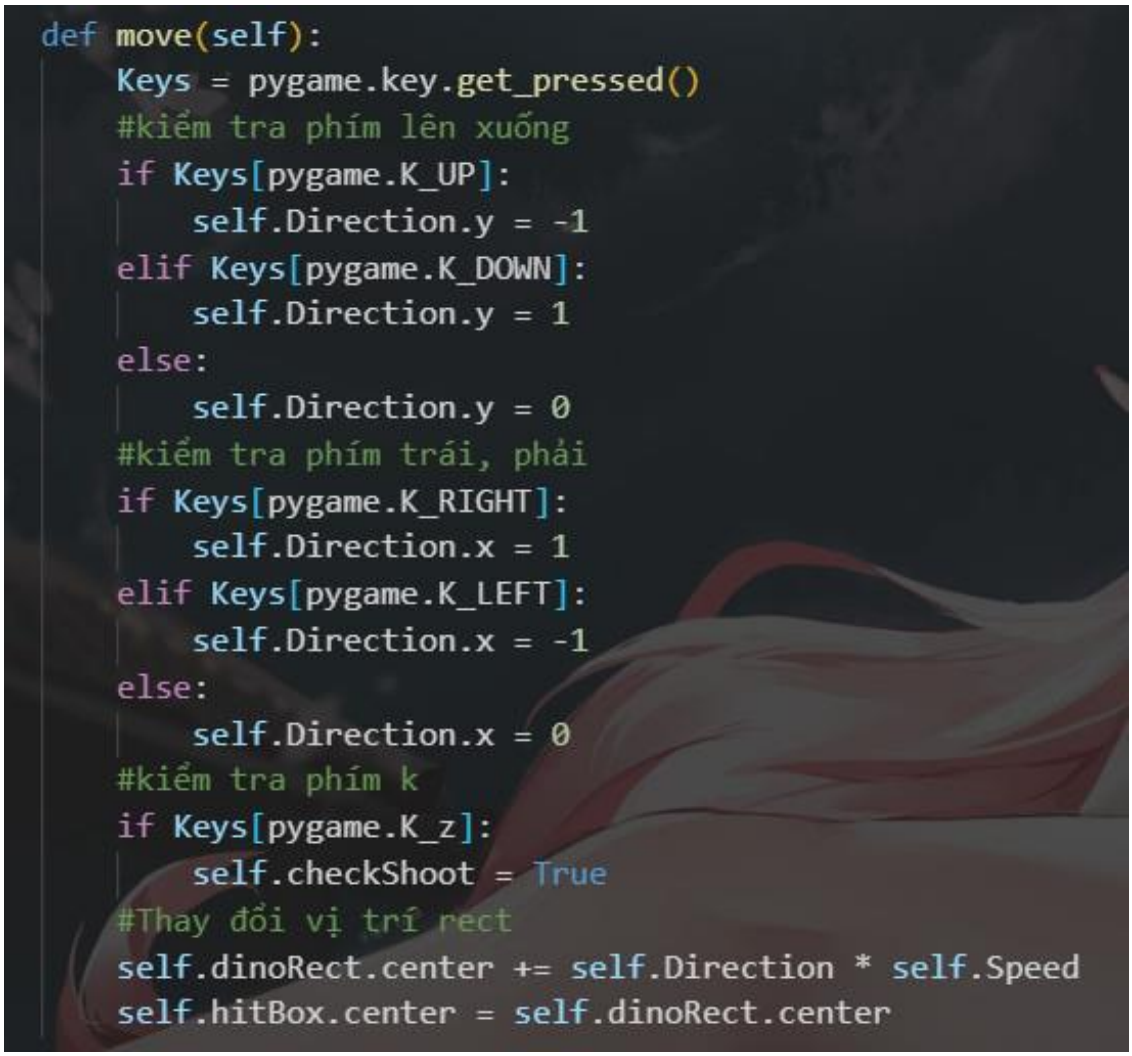
Khi khủng long chạm sàn, chạm nóc hoặc chạm ống thì màn 1 kết thúc, nếu số điểm lớn hơn hoặc bằng 100 thì người chơi có quyền lựa chọn chơi tiếp màn tiếp theo ngược lại người chơi chỉ được phép chơi lại hoặc thoát game, dữ liệu sẽ được lưu lại.

```
    #xử lý va chạm
    def check_collision(self):
        for Pipe in self.pipeList:
            if self.dinoRect.colliderect(Pipe):
                Sounds["hitSoundMap1"].play()
                return False
        if self.dinoRect.top <= -75 or self.dinoRect.bottom >= 380:
            return False
        return True
```

Màn 2:

```
def __init__(self, Surface):
    super().__init__()
    #Thiết lập hình ảnh
    self.Background = pygame.transform.scale(pygame.image.load(f"imgdino/map2/background.png").convert_alpha(), (Width, Height))
    self.dinoImages = []
    for i in range(6):
        self.dinoImages.append(pygame.transform.scale(pygame.image.load(f"imgdino/map2/dino{i}.png").convert_alpha(), (70, 80)))
    self.bossImage = pygame.transform.scale(pygame.image.load(f"imgdino/map2/boss.png").convert_alpha(), (130, 140))
    self.enemyImage = pygame.image.load(f"imgdino/map2/enemy.png").convert_alpha()
    self.bulletImage = pygame.transform.scale(pygame.image.load(f"imgdino/map2/bullet.png").convert_alpha(), (30, 30))
    self.Player = dinosauro_fly(self.dinoImages[0].get_rect(center = (100, 100)), self.bulletImage.get_rect, 5, 20)
    self.Boss = boss_fly(self.bossImage.get_rect(center = (700, 200)), self.enemyImage.get_rect(), 3, 100)
    self.dinoAnimation = 0
  #thiết lập thông tin game
    self.Score = 0
    self.highScore = 0
    self.Block = True
    self.bloodBossFont = pygame.font.Font(typeText, 30)
    self.playerFont = pygame.font.Font(typeText, 15)
    #tạo âm thanh
    Sounds["dinoShoot"] = pygame.mixer.Sound('imgdino/startgame/shoot1.wav')
    Sounds["bossCollide"] = pygame.mixer.Sound('imgdino/startgame/tick.wav')
    #tạo khung hình
    self.Surface = Surface
    self.Clock = pygame.time.Clock()
```

Sử dụng kỹ năng chuyển động linh hoạt giữa các ngón tay của người chơi, để tránh các vật thể từ boss đánh ra, đồng thời phản công lại boss bằng phím z



```
def move(self):
    Keys = pygame.key.get_pressed()
    #kiểm tra phím lên xuống
    if Keys[pygame.K_UP]:
        self.Direction.y = -1
    elif Keys[pygame.K_DOWN]:
        self.Direction.y = 1
    else:
        self.Direction.y = 0
    #kiểm tra phím trái, phải
    if Keys[pygame.K_RIGHT]:
        self.Direction.x = 1
    elif Keys[pygame.K_LEFT]:
        self.Direction.x = -1
    else:
        self.Direction.x = 0
    #kiểm tra phím k
    if Keys[pygame.K_z]:
        self.checkShoot = True
    #Thay đổi vị trí rect
    self.dinoRect.center += self.Direction * self.Speed
    self.hitBox.center = self.dinoRect.center
```

Boss có thể di chuyển và tấn công người chơi

```python
def update(self, numBullet):
    self.move()
    for Enemy in self.Enemys:
        Enemy.Rect.centerx -= Enemy.Speed
        Enemy.Rect.centery += Enemy.ranY
        if Enemy.Rect.centerx < 0:
            self.Enemys.remove(Enemy)
            numBullet += 1
        elif Enemy.Rect.centery < 0 or Enemy.Rect.centery > 450:
            self.Enemys.remove(Enemy)
    if numBullet > 5:
        numBullet = 5
    return numBullet

def create_enemy(self):
    e = enemy(self.bossRect.centerx, self.bossRect.centery,3)
    self.Enemys.append(e)
```

```python
def check_collide(self, Bullets):
    for Bullet in Bullets:
        if self.bossRect.colliderect(Bullet):
            self.Blood -= 10
            Sounds["bossCollide"].play()
            Bullets.remove(Bullet)
    return Bullets
```

Người chơi kiên nhẫn tránh những vật thể do boss đánh ra, đến khi có lượt tấn công, lúc đó người chơi sẽ phải phản công cho tới khi hạ được boss là qua màn

```python
def check_collide(self, Enemys):
    for Enemy in Enemys:
        if self.hitBox.colliderect(Enemy.Rect):
            self.Blood -= 5
            Enemys.remove(Enemy)
    return Enemys
```

```
if self.Boss.Blood <= 0:
    if self.Score > self.highScore:
        self.highScore = self.Score
    save_data(1, int(self.highScore), "f")
    checkStatus = announcement(True, self.Score, self.highScore, self.Surface)
```

Màn 3:

Các thuộc tính cần thiết cho màn chơi.

```
class map3():
    def __init__(self, Surface):
        super().__init__()
        #Thiết lập hình ảnh
        self.Background = pygame.transform.scale(pygame.image.load(f"imgdino/map3/background0.png"), (Width, Height))
        self.dinoImages = []
        for i in range(4):
            self.dinoImages.append(pygame.transform.scale(pygame.image.load(f"imgdino/map3/dino{i}.png"), (70, 73)))
        self.bossImages = []
        for i in range(3):
            self.bossImages.append(pygame.transform.scale(pygame.image.load(f"imgdino/map3/boss{i}.png"), (300, 300)))
        self.enemyImage = pygame.transform.scale(pygame.image.load(f"imgdino/map3/enemy.png"), (90, 50))
        self.bulletImage = pygame.image.load(f"imgdino/map3/bullet1.png")
        self.dinoAnimation = 0
        self.bossAnimation = 0
        #Tạo 2 đối tượng chính của màn chơi
        self.Player = dino_swim(self.dinoImages[0].get_rect(), 5, 1)
        self.Boss = boss_octopus(self.bossImages[0].get_rect(center = (1000, 250)), 3, 300)
        #thiết lập thông tin game
        self.Score = 0
        self.highScore = 0
        self.Block = True
        self.bossVisible = False
        self.bloodBossFont = pygame.font.Font(typeText, 30)
        self.playerFont = pygame.font.Font(typeText, 15)
```

```
#tạo âm thanh
Sounds["dinoShoot2"] = pygame.mixer.Sound('imgdino/startgame/shoot1.wav')
Sounds["bossHit"] = pygame.mixer.Sound("imgdino/startgame/hitboss.wav")
Sounds["ItemSound"] = pygame.mixer.Sound("imgdino/startgame/Item.wav")
#tạo khung hình
self.Surface = Surface
self.Clock = pygame.time.Clock()
```

Nhân vật sẽ chịu tác động của trọng lực rơi xuống dưới khi nhấn phím space nhân vật sẽ nổi lên

```python
class dino_swim():
    def __init__(self, dinoRect, jumpHeight, Blood):
        #tạo thuộc tính cho khủng long
        self.Blood = Blood
        self.jumpHeight = jumpHeight
        self.dinoRect = dinoRect
        self.hitBox = pygame.Rect(0,0, dinoRect.width - 50, dinoRect.height - 50)
        #tạo thuộc tính mảng viên đạn
        self.Bullets = []
        self.numBullet = 0
        #tạo trọng lực của khủng long
        self.Gravity = 0.3

    def update(self, bossRect, bossBlood):
        self.jumpHeight += self.Gravity
        self.dinoRect.centery += self.jumpHeight
        if self.dinoRect.y < 50:
            self.dinoRect.y = 50
        elif self.dinoRect.y >= 390:
            self.dinoRect.y = 390
        self.hitBox.center = self.dinoRect.center
```

Mỗi 1 giây sẽ có 1 con cá mập được sinh ra ở bên phải màn hình game và nó sẽ di chuyển theo hướng từ trái sang phải của màn hình game

```python
elif Event.type == self.eventBoss:
    self.Boss.create_enemy()
```

```python
def create_enemy(self):
    Enemy = pygame.Rect(850, random.randint(70, 370), 70, 73)
    self.Enemys.append(Enemy)
```

```python
def update1(self):
    for Enemy in self.Enemys:
        Enemy.x -= 5
        if Enemy.x < -50:
            self.Enemys.remove(Enemy)
            return 1
    return 0
```

Mỗi khi vượt qua 1 con cá mập sẽ được tăng 1 điểm, mỗi 15 sẽ nhận được 7 viên đạn

```
if self.Score  % 15 == 0 and self.Score != 0:
    Sounds["ItemSound"].play()
    self.Score += 1
    self.Player.numBullet += 7
```

Khi vượt qua mốc 17 điểm boss sẽ xuất hiện đồng thời toàn bộ cá mập trên màn hình đều biến mất

```
if self.Score > 17:
    self.bossVisible = True
    self.Boss.Enemys.clear()
    self.enemyImage = pygame.transform.scale(pygame.image.load f"imgdino/map3/bullet0.png") , (90, 50))
```

Mỗi 1 giây boss sẽ bắn 1 viên dạn

```
elif Event.type == self.eventBoss:
    self.Boss.create_enemy()
```

Khi nhấn phím E trên bàn phím nếu có đạn nhân vật sẽ bắn 1 thanh kiếm theo hướng từ trái sang phải của màn hình game

```
elif Event.type == pygame.KEYDOWN:
    if Event.key == pygame.K_e:
        if self.Player.numBullet > 0:
            Sounds["dinoShoot2"].play()
            self.Player.create_bullet()
            self.Player.numBullet -= 1
```

```
def create_bullet(self):
    Bullet = pygame.Rect(850, random.randint(70, 370), 50, 20)
    Bullet.center = self.dinoRect.center
    self.Bullets.append(Bullet)
```

```
for Bullet in self.Bullets:
    if Bullet.colliderect(bossRect):
        Sounds["bossHit"].play()
        bossBlood -= 10
        self.Bullets.remove(Bullet)
    elif Bullet.x > Width:
        self.Bullets.remove(Bullet)
    else:
        Bullet.x += 7
```

Khi nhân vật chạm vào cá mập hoặc viên đạn của boss thì màn chơi kết thúc

```python
def update2(self, dinoRect):
    if self.bossRect.centerx > 700:
        self.bossRect.centerx -= 5
        self.hitBox.centerx -= 5
    for Enemy in self.Enemys:
        Enemy.x -= 5
        if Enemy.colliderect(dinoRect):
            return 0, True
        elif Enemy.x < -50:
            self.Enemys.remove(Enemy)
            return 1, False
    return 0, False

def update1(self, dinoRect):
    for Enemy in self.Enemys:
        if Enemy.colliderect(dinoRect):
            return 0, True
        Enemy.x -= 5
        if Enemy.x < -50:
            self.Enemys.remove(Enemy)
            return 1, False
    return 0, False
```

```python
if self.bossVisible:
    core, checkLose = self.Boss.update2(self.Player.hitBox)
    self.Score += core
else:
    core, checkLose = self.Boss.update1(self.Player.hitBox)
    self.Score += core
```

```python
elif checkLose:
    if self.Score > self.highScore:
        self.highScore = self.Score
    save_data(2, int(self.highScore), "f")
    checkStatus = announcement(False, self.Score, self.highScore, self.Surface)
    if checkStatus == (True, True):
        self.reset_game()
        checkLose = False
        continue
```

Khi bắn trúng boss máu boss sẽ giảm đi 10 giảm về 0 người chơi sẽ chiến thắng

```
if self.Boss.Blood <= 0:
    if self.Score > self.highScore:
        self.highScore = self.Score
    save_data(2, int(self.highScore), "f")
    checkStatus = announcement(True, self.Score, self.highScore, self.Surface)
    if checkStatus == (True, True):
        self.reset_game()
        continue
    elif checkStatus == (False, False):
        pygame.time.set_timer(self.eventPlayer, 0)
        pygame.time.set_timer(self.eventBoss, 0)
        return False
    else:
        pygame.time.set_timer(self.eventPlayer, 0)
        pygame.time.set_timer(self.eventBoss, 0)
        return True
```

```
def update(self, bossRect, bossBlood):
    self.jumpHeight += self.Gravity
    self.dinoRect.centery += self.jumpHeight
    if self.dinoRect.y < 50:
        self.dinoRect.y = 50
    elif self.dinoRect.y >= 390:
        self.dinoRect.y = 390
    self.hitBox.center = self.dinoRect.center
    for Bullet in self.Bullets:
        if Bullet.colliderect(bossRect):
            Sounds["bossHit"].play()
            bossBlood -= 10
            self.Bullets.remove(Bullet)
        elif Bullet.x > Width:
            self.Bullets.remove(Bullet)
        else:
            Bullet.x += 7
    return bossBlood
```

Khi nhấn replay trên màn hình thông báo sẽ được chơi lại màn đó

```
def reset_game(self):
    self.Score = 0
    self.bossVisible = False
    self.enemyImage = pygame.transform.scale(pygame.image.load(f"imgdino/map3/enemy.png"), (90, 50))
    self.Boss.reset_game()
    self.Player.reset_game()
```

```
def reset_game(self):
    self.bossRect.centerx = 1000
    self.hitBox.x = self.bossRect.x + 150
    self.Blood = 300
    self.Enemys.clear()
```

```
def reset_game(self):
    self.Blood = 1
    self.Bullets.clear()
    self.numBullet = 0
    self.dinoRect.topleft = (0, 0)
```

- **Tạo hình nhân vật, quái vật, ngữ cảnh xung quanh nhân vật**

Nhân vật

Màn 1:

```
#thiết lập hình ảnh
self.Background = pygame.transform.scale(pygame.image.load(f"imgdino\\map1\\background0.png").convert_alpha(),
self.floorXPos = 0
self.Floor = pygame.transform.scale2x(pygame.image.load("imgdino/map1/background3.png").convert())
self.dinoDown = pygame.image.load('imgdino/map1/dino-fly-down.png')
self.dinoMid = pygame.image.load('imgdino/map1/dino-fly.png')
self.dinoUp = pygame.image.load('imgdino/map1/dino-fly-up.png')
self.dinoList= [self.dinoDown, self.dinoMid, self.dinoUp]
self.dinoIndex = 0
self.Dinosaur = self.dinoList[self.dinoIndex]
self.dinoRect = self.Dinosaur.get_rect(center = (200, 100))
```

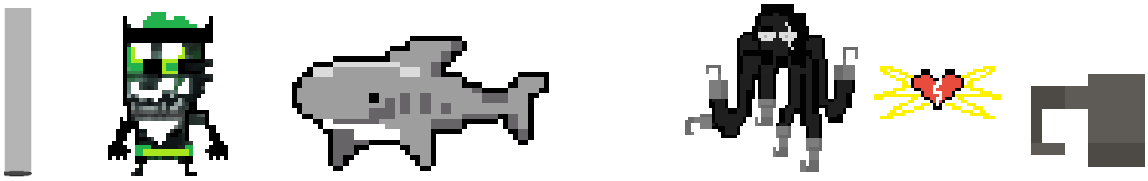
Màn 2:

```
#Thiết lập hình ảnh
self.Background = pygame.transform.scale(pygame.image.load(f"imgdino/map2/background.png").convert_alpha(), (Width, Height))
self.dinoImages = []
for i in range(6):
    self.dinoImages.append(pygame.transform.scale(pygame.image.load(f"imgdino/map2/dino{i}.png").convert_alpha(), (70, 80)))
self.bossImage = pygame.transform.scale(pygame.image.load(f"imgdino/map2/boss.png").convert_alpha(), (130, 140))
self.enemyImage = pygame.image.load(f"imgdino/map2/enemy.png").convert_alpha()
self.bulletImage = pygame.transform.scale(pygame.image.load(f"imgdino/map2/bullet.png").convert_alpha(), (30, 30))
self.Player = dinosauro_fly(self.dinoImages[0].get_rect(center = (100, 100)), self.bulletImage.get_rect, 5, 20)
self.Boss = boss_fly(self.bossImage.get_rect(center = (700, 200)), self.enemyImage.get_rect(), 3, 100)
```

Màn 3:

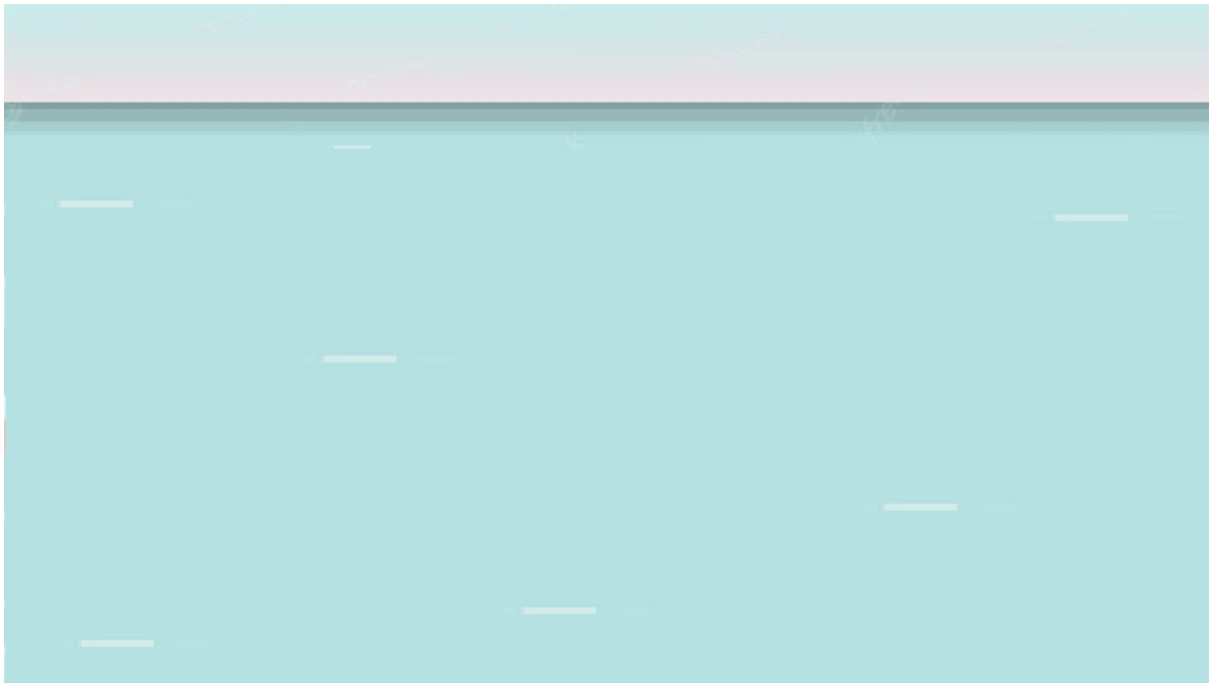
```
#Thiết lập hình ảnh
self.Background = pygame.transform.scale(pygame.image.load(f"imgdino/map3/background0.png"), (Width, Height))
self.dinoImages = []
for i in range(4):
    self.dinoImages.append(pygame.transform.scale(pygame.image.load(f"imgdino/map3/dino{i}.png"), (70, 73)))
self.bossImages = []
for i in range(3):
    self.bossImages.append(pygame.transform.scale(pygame.image.load(f"imgdino/map3/boss{i}.png"), (300, 300)))
self.enemyImage = pygame.transform.scale(pygame.image.load(f"imgdino/map3/enemy.png"), (90, 50))
self.bulletImage = pygame.image.load(f"imgdino/map3/bullet1.png")
```

Quái vật (vật cản)



Vũ khí của nhân vật

- **Thiết kế background**



7. ***Kết luận***

- **Trình bày ngắn gọn kết quả của dự án**
  + có cốt truyện, nội dung rõ ràng
  + dự án đạt đầy đủ các yêu cầu đề ra
  + đảm bảo về mặt hình thức, kiến trúc và thẩm mỹ
  + kết hợp tốt các công cụ hỗ trợ
  + xây dựng thành công dự án game
- **Kinh nghiệm và bài học rút ra trong quá trình phát triển game**
  + bổ sung kiến thức về ngôn ngữ lập trình python, game
  + luyện khả năng tư duy logic
  + nâng cao kỹ năng nhóm
  + xây dựng kỹ năng cơ bản về lập trình và phát triển game bằng ngôn ngữ lập trình python

8. ***Hướng phát triển tiếp theo***

- **Mở rộng nội dung**
  + Tạo thêm màn chơi
  + Xây dựng thêm cốt truyện
  + Tăng độ thích thú cho trò chơi
  + Thêm chức năng
  + Thêm hiệu ứng đặc biệt cho game

- **Tối ưu hóa**

  + Loại bỏ các đoạn mã dư thừa

  + Thêm các tính năng giúp game vận hành ổn định hơn

  + Cải tiến đồ họa

  + Tối ưu hiệu suất

  + Tăng tốc độ trò chơi

- **Đa nền tảng**

  + Vận hành trên các nền tảng ios, android, java, linux, micorosoft window phone, window,…

- **Đa thể loại**

  + Mở rộng các màn chơi

- **Cải thiện hệ thống quản lý**

  + Liên kết database

  + Mã hóa dữ liệu

  + Quản lý tài khoản của người chơi

  + Tăng cường tính năng an ninh

  + Tích hợp phân tích dữ liệu

- **Mở rộng thế giới của trò chơi**

  + Phát triển tinh cách cho nhân vật

  + Mở rộng game bằng các game liên quan

  + Thêm nhân vật mới

  + Thêm ngữ cảnh trong trò chơi

  + Thêm các nhiệm vụ cho nhân vật thực hiện

9. ***Tham khảo***

- Pygame.org.docs
- Youtube
- Web3School
- GitHub
- Pixel Art
- https://mixkit.co/