BÀI SỐ 5. GAME: CHIẾN BINH KHÔNG GIAN 1

- **Kiến thức:**

- Ôn lại kiến thức về hàm
- Chức năng hiển thị ảnh nền, hình ảnh nhân vật của Turtle
- Điều khiển nhân vật bằng các phím di chuyển
- Thiết kế kịch bản Game

- **Kỹ năng:**

- Làm việc với ngôn ngữ python, kĩ năng làm việc nhóm.
- Luyện tập kỹ năng thuyết trình qua việc ôn tập cuối buổi học.

- **Sản phẩm:** Chương trình Game "**Chiến binh không gian"**
- **Phân bổ thời gian:**

Nội dung	Thời gian
1. Chèn hình nền, ảnh nhân vật **2.** Sử dụng các phím điều khiển nhân vật	**45'**
3. Thiết kế game: chiến binh không gian 1 + Hướng dẫn + Làm bài + Thuyết trình **4. Tổng kết đánh giá**	**1h15'**

1. Chèn hình ảnh nền, hình ảnh nhân vật trong Turtle

Cô ơi! Thư viện đồ họa Turtle chỉ cho phép khởi tạo được các hình ảnh: 

Vậy, em muốn chèn thêm ảnh nền khác được không ạ?

Thư viện đồ họa Turtle không chỉ cho phép chúng ta khởi tạo các hình như trên, mà còn cho phép các em khởi tạo các hình ảnh đối tượng, các hình nền khác nhau. Các em hãy xem ví dụ 1 sau nhé:

Ví dụ 1. Chèn hình nền, đối tượng đồ họa

```
# Chèn thư viện đồ họa turtle
import turtle
# Khởi tạo màn hình đồ họa
screen = turtle.Screen()
# thiết lập độ rộng của màn hình đồ họa
screen.setup(400, 400)
#thiết lập hình ảnh nền cho màn hình đồ họa
screen.bgpic("space.gif")
# Gán biến image
image = "tau.gif"
# Chèn đối tượng hình ảnh vào màn hình đồ họa
screen.addshape(image) turtle.shape(image)
```

+ Chèn ảnh nền, ta sử dụng hàm: **bgpic(ảnh.gif)**

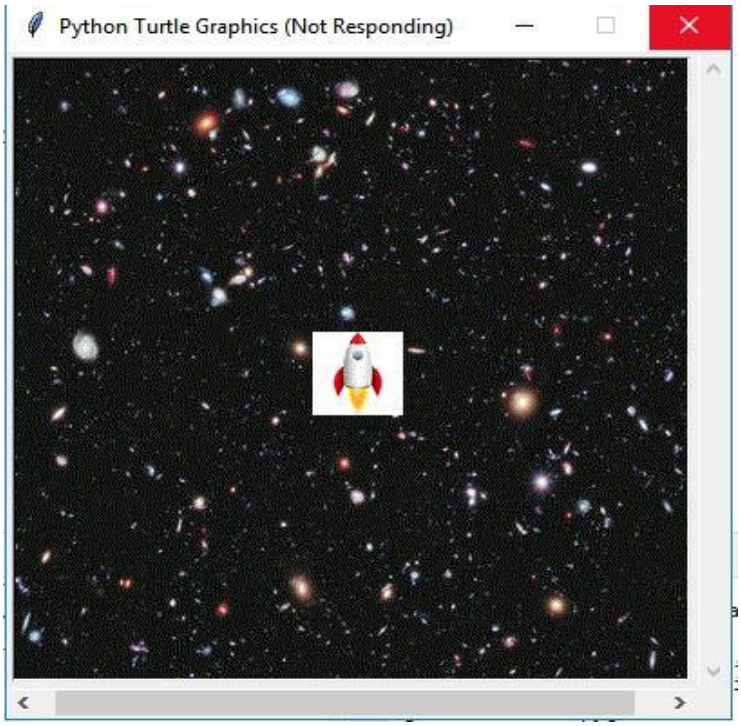
<screen>.**bgpic("ảnh.gif ")**

+ Chèn hình ảnh nhân vật vào màn hình nền:

```
<screen>.addshape(ảnh.gif)
turtle.shape(ảnh.gif)
```

Chú ý: Turtle chỉ hỗ trợ file ảnh có đuôi .gif

Kết quả của đoạn mã trên:



(Hình 1.1. hình nền và ảnh đối tượng đồ họa được chèn thêm vào)

Trong bài số 4, các em đã biết cách lập trình điều khiển đối tượng đồ họa di chuyển theo các hướng khác nhau thông qua các hàm: **forward(), turnright()** và hàm **turnleft().** Turtle còn cung cấp các hàm giúp ta điều khiển các đối tượng đồ họa bằng các phím. *Để tìm hiểu vấn đề này, chúng ta chuyển sang phần tiếp theo nhé!*

2. Điều khiển nhân vật bằng các phím

Để điều khiển hướng di chuyển của nhân vật, Turtle chung cấp cho chúng ta hàm **Onkey(),** trước khi tìm hiểu về hàm này chúng ta xét ví dụ 02:

Ví dụ 2. Di chuyển nhân vật bằng các phím

```
# Chèn thư viện đồ họa turtle
import turtle
# Khởi tạo màn hình đồ họa
screen = turtle.Screen()
# thiết lập độ rộng của màn hình đồ họa
screen.setup(400, 400)
#thiết lập hình ảnh nền cho màn hình đồ họa
screen.bgpic("space.gif")
# thiết lập hình ảnh nền cho màn hình đồ họa
turtle.shape(“triangle”)
turtle.color("white")
# thiết lập tốc độ di chuyển của nhân vật
move_speed = 10
turn_speed = 10


#  thiết lập các hàm
def forward():
  turtle.forward(move_speed)
def backward():
  turtle.backward(move_speed)
def left():
  turtle.left(turn_speed)
def right():
  turtle.right(turn_speed)
```

```
# thiết lập không vẽ khi di chuyển
turtle.penup()
# hàm onkey: sẽ gọi các hàm khi bạn nhấn vào các phím tương ứng
screen.onkey(forward, "Up")
screen.onkey(backward, "Down")
screen.onkey(left, "Left")
screen.onkey(right, "Right")
screen.listen()
```

Kết quả: khi ta nhấn các phím **Up**, **Down**, **Left** hoặc **Right** nhân vật sẽ di chuyển và quay sang các hướng tương ứng. Các em hoàn toàn có thể sử dụng bất kỳ các phím khác, chỉ cần thiết lập lại trong code là được, ví dụ:

screen.onkey(forward, "**w**")

screen.onkey(backward, "**s**")

screen.onkey(left, "**a**")

screen.onkey(right, "**d**")

screen.listen()

Hàm onkey() của đối tượng đồ họa có hai tham số đó là hàm gọi và phím nhấn: **onkey(<hàm ABC>,< phím XYZ>):** Hàm ABC sẽ được gọi khi ta nhấn phím XYZ. Khi các em nhấn phím XYZ thì sẽ mọi công việc trong hàm ABC sẽ được triệu gọi.

3. Thiết kế Game số 1: Chiến binh không gian

Trong bài học ngày hôm nay, chúng ta sẽ làm 1 Game đơn giản với nội dung như sau: Một quái vật đang xâm chiếm trái đất; 1 siêu anh hùng có nhiệm vụ bảo vệ trái đất bằng cách tiêu diệt quái vật (*tiêu diệt bằng cách lao vào quái vật để quái vật nổ tung*).

Kịch bản Game:

- Nhân vật: Có 2 nhân vật: quái vật (*là 1 hình ảnh.gif)* và siêu anh hùng (chính là hình con rùa –turtle)
- Di chuyển:

+ Quái vật sẽ di chuyển trong vùng đồ họa, nếu va chạm biên vùng đồ họa sẽ bật lại di chuyển ngược lại

+ Siêu anh hùng sẽ được người chơi điều khiển để lao vào quái vật; nếu lao trúng thì trò chơi kết thúc và giải cứu được thế giới.

- Trò chơi kết thúc khi: Siêu anh hùng lao được vào quái vật

3.1. Tạo nhân vật siêu anh hùng, nền và vẽ khung giới hạn di chuyển

```
import turtle
import random
# thiết lập màn hình đồ họa
sc = turtle.Screen()
#thiết lập hình ảnh nền cho màn hình đồ họa
sc.bgpic("space.gif")
```

 # Vẽ vùng giới hạn di chuyển:

```
mypen = turtle.Turtle()
mypen.penup()
mypen.setposition(-300, -300)
mypen.pendown()
mypen.pensize(3)
mypen.speed(0)
for i in range(4):
    mypen.forward(600)
    mypen.left(90)
# vẽ xong, ẩn đối tượng vẽ
mypen.hideturtle()
```

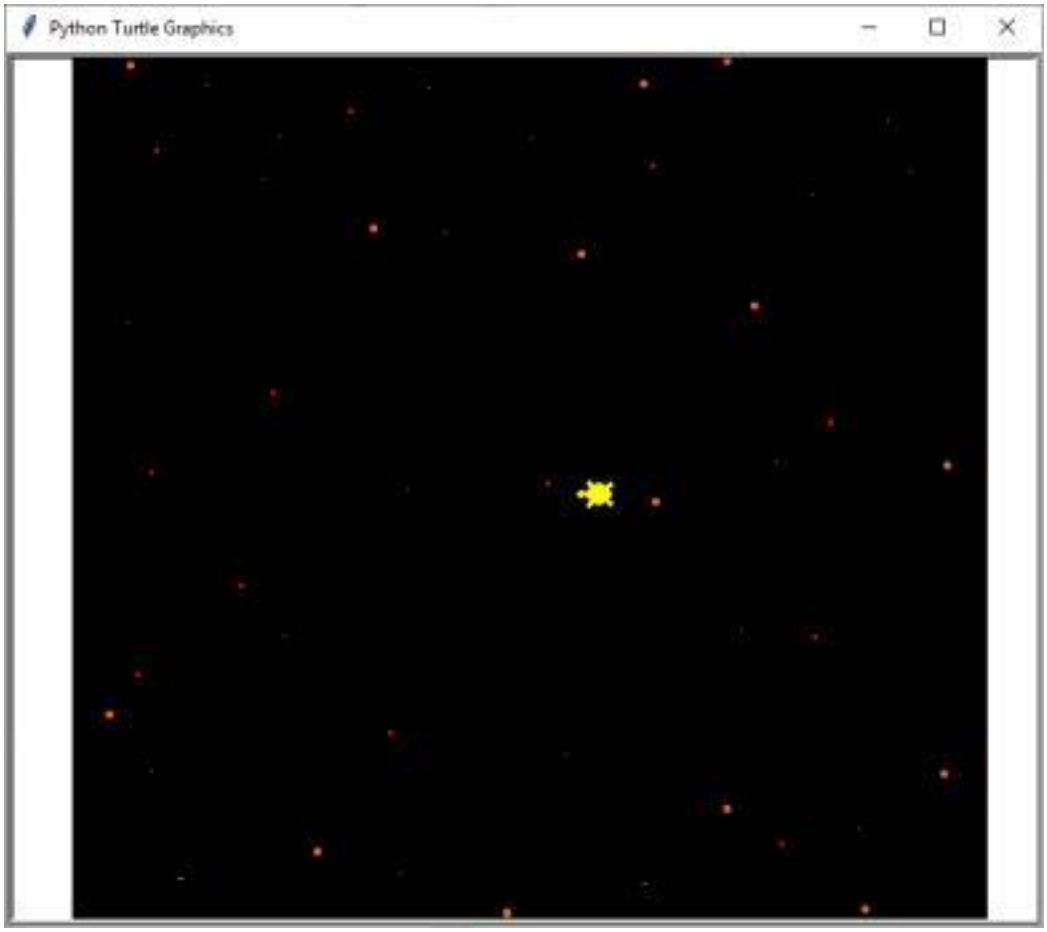
 # Tạo ra nhân vật siêu anh hùng

```
player = turtle.Turtle()
player.color("yellow")
player.shape("turtle")
player.penup()
# thiết lập tốc độ siêu anh hùng
speed = 1
# định nghĩa các phím ra chuyển
def turnleft():
    player.left(30)
def turnright():
    player.right(30)
```

```
def increaseSpeed():
    global speed
    speed += 1
    if speed >= 5:
        speed = 5
def decreaseSpeed():
    global speed
    speed -= 1
    if speed <= -5:
        speed = -5


# khi nhấn phím
turtle.listen()
turtle.onkey(turnleft, "Left")
turtle.onkey(turnright, "Right")
turtle.onkey(increaseSpeed, "Up")
turtle.onkey(decreaseSpeed, "Down")
# Xử lý khi siêu anh hùng di chuyển va chạm vào
biên while True:
    # thiết lập siêu anh hùng tiến về phía trước
  player.forward(speed)
    #kiểm tra xem khi di chuyển có chạm biên không
    if player.xcor() < -290 or player.xcor() > 290:
        player.right(180)
    if player.ycor() < -290 or player.ycor() > 290:
        player.right(180)
```

Khi chạy chương trình, kết quả sẽ ra như sau:



3.2. Bổ sung thêm quái vật và xử lý tình huống va chạm

Chú ý: Để xác định khoảng cách giữa hai điểm A và B trong tọa độ (X, Y), ta áp dụng công thức toán học:

$$d = \sqrt{(\Delta x)^2 + (\Delta y)^2}, \text{ với } (\Delta x = x_A - x_B, \Delta y = y_A - y_B)$$

Ta bổ sung thêm code sau vào code phần 3.1 ở trên:

```
# Tạo ra quái vật
goal = turtle.Turtle()
image="quaivat.gif"
sc.addshape(image)
```

```
        goal.shape(image)
        goal.penup()
        goal.speed(0)
        # hàm setposition() đặt đối tượng ở vị trí (x, y)
        goal.setposition(random.randint(-300, 300), random.randint(-300, 300))
```

 # Kiểm tra va chạm giữa hai nhân vật

```
def isCollision(t1, t2):
    # khoảng cách giữa siêu anh hùng và quái vật
    d = math.sqrt(math.pow(t1.xcor() - t2.xcor(),2) + math.pow(t1.ycor() - t2.ycor(),2))
    if d<20:
        return True
    else:
        return False
```

 # Tạo hàm kiểm tra các nhân vật khi chạm biên

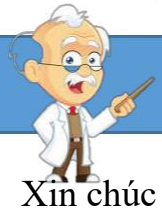
```
def boundaryChecking(t):
    # hàm xcor(): lấy tọa độ X hiện tại của đối tượng, ycor(): lấy tọa độ y
    if t.xcor() < -300 or t.xcor() > 300:
        t.right(180)
        t.setposition(random.randint(-300, 300), random.randint(-300, 300))
    if t.ycor() < -300 or t.ycor() > 300:
        t.right(180)
        t.setposition(random.randint(-300, 300), random.randint(-300, 300))
```

 # Viết lại phần

while while True:

```
    player.forward(speed)
    # kiểm tra siêu anh hùng có chạm biên hay không
    boundaryChecking(player)
     # kiểm tra va chạm
    if isCollision(player, goal):
        print("Đã giải cứu được thế giới")
        break
    # thiết lập quái vật di chuyển
    goal.forward(10)
    goal.left(random.randint(10,180))
    # kiểm tra quái vật có chạm biên hay
    không boundaryChecking(goal)
```

Kết quả khi chạy như sau:



TỔNG KẾT BÀI HỌC

Xin chúc mừng các em đã hoàn thành bài học số 5, trong bài này, các em cần chú ý những nội dung sau:

1. **Chèn hình nền, chèn nhân vật đồ họa:**

   + Chèn ảnh nền, ta sử dụng hàm: **bgpic(ảnh.gif)**

   <screen>.**bgpic("ảnh.gif ")**

   + Chèn hình ảnh nhân vật vào màn hình nền:

   ```
   <screen>.addshape(ảnh.gif)
   turtle.shape(ảnh.gif)
   ```

2. **Sử dụng phím để điều khiển đối tượng**

   ```
   screen.onkey(<hàm>, "<phím>")
   screen.listen()
   ```

3. **Một số hàm khác của đối tượng đồ hòa:**

   + Hàm xcor(): lấy tọa độ X hiện tại của đối tượng,

   + Hàm ycor(): lấy tọa độ y

   + Hàm setposition() đặt đối tượng đồ họa ở vị trí (x, y)

Nội dung bài học tiếp theo: Chúng ta sẽ phát triển game: chiến binh không gian với nhiều tính năng hơn: nhạc, trang bị vũ khí cho các đối tượng, thông báo điểm khi bắn trúng,…

BÀI TẬP VỀ NHÀ

1. Chuẩn bị thêm các hình nền để có thể chuyển cảnh và thêm vào chương trình (chú ý ảnh có định dạng đuôi file .gif)
2. Chuẩn bị thêm âm thanh va chạm, tiếng súng,…
3. Thêm đa dạng các quái vật xuất hiện ở nhiều vị trí khác nhau
4. Phát triển ý tưởng phong phú cho game: chiến binh không gian

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..