Introdução ao Pygame

Criando o Brick Breaker III

Python para Todos

CEFET-MG

Conteúdo

Criação do restante de blocos

Alinhamento correto dos blocos

Colisão com blocos

Criação da função move bola

Verificar se perdeu o jogo

Verificar se venceu o jogo

Criação do restante de blocos

Com a classe “Bloco” criada, iremos fazer com que vários blocos irão aparecer na tela. Para isso:

- Criaremos uma outra classe dentro do arquivo da classe Bloco chamada Blocos que será composta de vários blocos
- ele receberá a quantidade de blocos (linha e colunas) que ele terá no total

Criação do restante de blocos

```
# No final da classe BLOCO

class Blocos:

  def __init__(self,  cor):
    self.cor = cor
    self.blocos = []
    self.espacamento = 4
```

Criação do restante de blocos

```
def cria_blocos(self,
 numero_de_linhas,numero_de_colunas):

  for i in range(numero_de_linhas):
    for j in range(numero_de_colunas):
      bloco= Bloco(self.cor)

      bloco.cria_bloco(
        j * (70+ self.espacamento),
        i * (70+ self.espacamento),
        70, 70
      )
      self.blocos.append(bloco)
```

Criação do restante de blocos

```
# SETUP
numero_de_linhas=5
numero_de_colunas=8
blocos = Blocos(("green"))
blocos.cria_blocos(numero_de_linhas, numero_de_colunas)
```

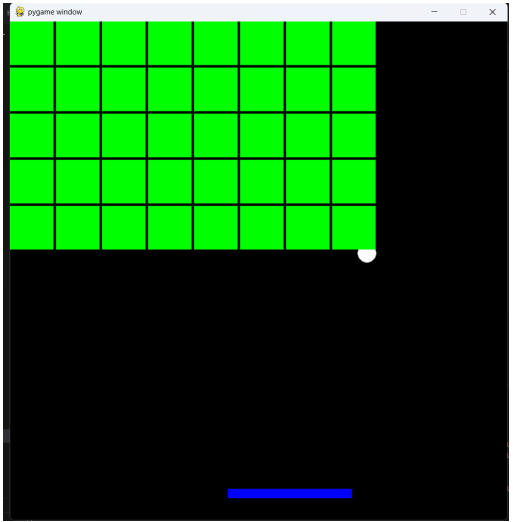
Criação do restante de blocos

```
# LOOP PRINCIPAL
  desenha_elementos(tela, jogador, bola, blocos)
```

Criação do restante de blocos

```
def desenha_elementos(tela, jogador, bola, blocos):
  #...
  for bloco in blocos.blocos:
    pg.draw.rect(tela, bloco.cor, bloco.bloco)
```

Criando a tela



Conteúdo

Criação do restante de blocos

Alinhamento correto dos blocos

Colisão com blocos

Criação da função move bola

Verificar se perdeu o jogo

Verificar se venceu o jogo

Alinhamento correto dos blocos

Iremos alinhar os blocos com o tamanho da tela. Mesmo se mudar o tamanho da tela, os blocos:

- Vão manter o alinhamento
- Vão manter o espaçamento

Alinhamento correto dos blocos

```
# Na classe Blocos, def cria_blocos
self.espacamento = int(
  largura/(numero_de_colunas*20.0)
)

espaco_para_os_blocos = (
  largura - self.espacamento * numero_de_colunas
)

bloco_largura = int(
  espaco_para_os_blocos / numero_de_colunas
)

bloco_altura = 0.3 * altura / numero_de_linhas
```

Alinhamento correto dos blocos

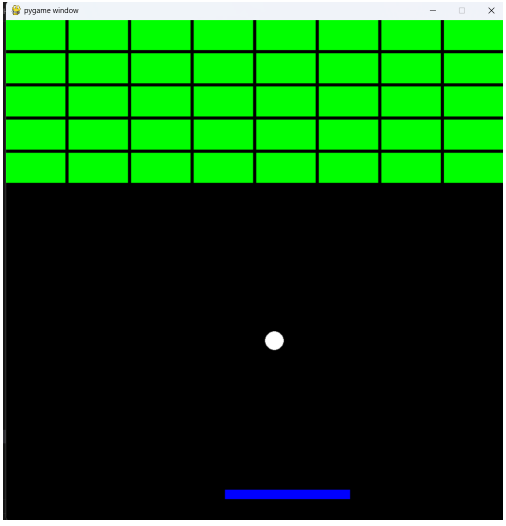
```
# Na classe Blocos
# Dentro da def cria_blocos
# Substituir a chamada da função cria_bloco por:
bloco.cria_bloco(
  j * (bloco_largura + self.espacamento),
  i * (bloco_altura  + self.espacamento),
  bloco_largura,
  bloco_altura
)
```

Alinhamento correto dos blocos

```
# Na classe brick_breaker
# Dentro da função inicia_jogo()
# acrescentar largura e altura como parametro

blocos.cria_blocos(
  numero_de_linhas,
  numero_de_colunas,
  largura,
  altura
)
```

Criando a tela



Conteúdo

Criação do restante de blocos

Alinhamento correto dos blocos

Colisão com blocos

Criação da função move bola

Verificar se perdeu o jogo

Verificar se venceu o jogo

Colisão com blocos

Iremos montar a colisão com o bloco com a bola.

- Quando a bola colidir com um bloco, o bloco terá que ser removido
- Além disso, no momento da colisão, a velocidade y deverá ser invertida

Colisão com blocos

```
def confere_colisao_blocos(self, blocos):

  for bloco in blocos.blocos:

    if self.bola.colliderect(bloco.bloco):
      blocos.blocos.remove(bloco)
```

Colisão com blocos

```
# Ainda dentro da funcao confere_colisão
#...

if(self.velocidade[1]<0):
  self.velocidade[1] = self.velocidade_y

elif(self.velocidade[1]>0):
  self.velocidade[1] = -self.velocidade_y
```

Colisão com blocos

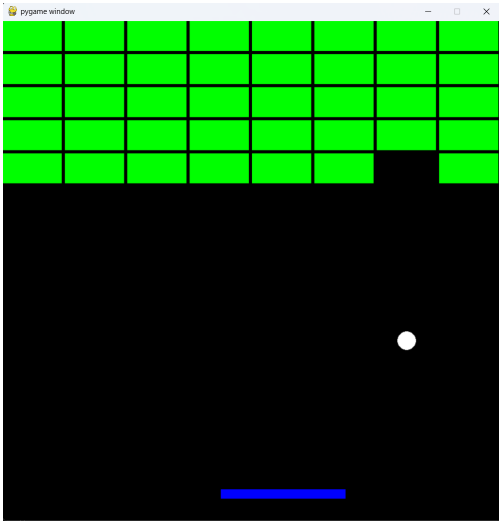
```
  # Ainda classe Bola
  # Dentro da funcao def move
def move(self, jogador, largura, altura, blocos):
  self.confere_colisao_blocos(blocos)
```

Colisão com blocos

```
# Na classe brick_breaker
# Dentro da funcao inicia_jogo
# Dentro do while
# Acrescentar blocos como parametro de bola.move

bola.move(jogador, largura, altura, blocos)
```

Criando a tela



Conteúdo

Criação do restante de blocos

Alinhamento correto dos blocos

Colisão com blocos

Criação da função move bola

Verificar se perdeu o jogo

Verificar se venceu o jogo

Criação da função move bola

```
# Na classe brick_breaker
# Criar a seguinte função

def move_bola(
  bola,
  jogador,
  largura,
  altura,
  blocos
):
  bola.move(jogador, largura, altura, blocos)
```

Criação da função move bola

```
# Ainda na classe brick_breaker
# Dentro da funcao inicia_jogo
# Dentro do while
# Trocar as linhas:

# bola.move(jogador, largura, altura, blocos)

# Por
move_bola(bola, jogador, largura , altura, blocos)
```

Conteúdo

Criação do restante de blocos

Alinhamento correto dos blocos

Colisão com blocos

Criação da função move bola

Verificar se perdeu o jogo

Verificar se venceu o jogo

Verificar se perdeu o jogo

Quando a bola atingir a região da parte debaixo da tela, o jogador perde, então:

- O tela irá congelar
- Aparecerá a mensagem de perdedor

Verificar se perdeu o jogo

```
def confere_colisao_parede(self, largura, altura):
  # no final da função confere_colisao_parede
  #...
  if self.bola.y >= altura-self.tamanho_bola:
    return False
  return True
```

Verificar se perdeu o jogo

```
def move(self, jogador, largura, altura, blocos):
  # no final da função
  return self.confere_colisao_parede(largura, altura)
```

Verificar se perdeu o jogo

```
# Na classe principal
# SETUP
estado = 0

  # LOOP PRINCIPAL
  estado = move_bola(bola,
                     jogador,
                     largura,
                     altura,
                     blocos
                    )
```

Verificar se perdeu o jogo

```
# Na classe principal
def move_bola(bola, jogador, largura, altura, blocos):
  if(bola.move(jogador,largura, altura, blocos)):
    return 0
  return 1  # Jogo acaba
```

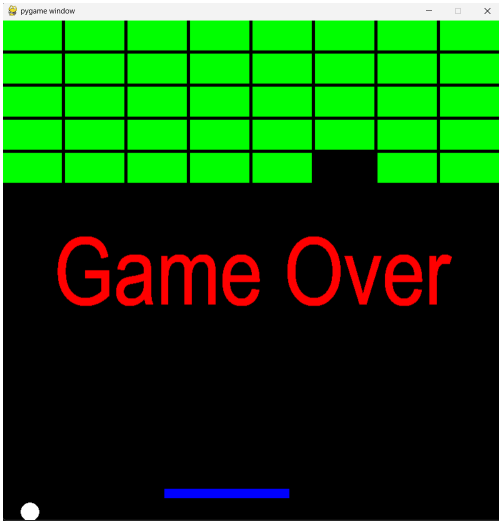
Verificar se perdeu o jogo

```
# Na classe brick_breaker
def desenha_elementos(tela, jogador, bola, blocos,
   estado):
  if(estado == 1): # Se o jogo acabou
    fonte = pg.font.SysFont("Arial", 150)
    texto = fonte.render("Game Over", True, "red")
    tela.blit(
      texto,
      (
        tela.get_width()//2 - texto.get_width()//2,
        tela.get_height()//2 - texto.get_height()//2
      )
    )
```

Verificar se perdeu o jogo

```
else:
  # O resto dentro de um else, menos update
```

Criando a tela



Conteúdo

Criação do restante de blocos

Alinhamento correto dos blocos

Colisão com blocos

Criação da função move bola

Verificar se perdeu o jogo

Verificar se venceu o jogo

Verificar se venceu o jogo

Por último, quando todos o blocos acabarem, o jogador vence.

- O jogo irá congelar
- Aparecerá a mensagem para o vencedor

Verificar se venceu o jogo

```
def move_bola(bola, jogador, largura, altura, blocos):
  if(blocos.blocos.__len__() == 0):
    return 2
  if(bola.move(jogador, largura, altura, blocos)):
    return 0
  return 1
```

Verificar se venceu o jogo

```
# Na classe brick_breaker
# Dentro da função desenha_elementos:
# apos o termino do if
elif(estado == 2):  # Se venceu
  fonte = pg.font.SysFont("Arial", 150)
  texto = fonte.render("You Win", True, "green")
  tela.blit(
    texto,
    (
      tela.get_width() // 2 - texto.get_width() // 2,
      tela.get_height() // 2 - texto.get_height() // 2
    )
  )
```

Criando a tela