Wormy - Guia passo passo de como implementar seu próprio jogo

Passo 1: Criar uma janela vazia

- Esse é o código base para o jogo. Não é necessário nenhum código extra para completar esse passo.

```
#### 1 - Criar janela vazia e loop principal do jogo #####

import pygame
from pygame.locals import *

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')
while True:
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()
    pygame.display.update()
```

Passo 2: Adicionar a cobrinha estática

Nesse passo deve-se:

- Criar a cobrinha.
- Desenhar a cobra na tela

A cobrinha é um conjunto de coordenadas (x,y). Você pode decidir qual vai ser o tamanho de cada “pixel” ou cada pedacinho da cobra. Nesse tutorial iremos utilizar quadradinhos 10x10. Por exemplo, a cobrinha pode iniciar com 3 segmentos nas posições: [(200,200),(210,200)(220,200)].

Cada quadrinho da cobra é representado por um Surface do pygame.

Dicas para esse passo:

- Para criar o quadrinho que representa um segmento da cobra pode-se criar uma Surface do pygame. Esse objeto recebe em seu construtor seu tamanho (largura,altura).
- O objeto Surface tem o método *fill* que recebe como argumento uma sequência RGB (a cor branca é representada pela sequência 255,255,255).
- Antes de desenhar qualquer coisa na tela é importante limpar a mesma. O objeto display do pygame possui um método *fill* que pode receber (0,0,0) para limpar a tela.
- Para desenhar a cobrinha deve-se iterar por todas as duas coordenadas e utilizar o método *blit(surface,pos)* do display para desenhar a pele da cobrinha (Surface) em uma determinada posição.

Passo 3: Fazer a cobrinha se movimentar

Para fazer com que a cobrinha se movimente deve-se modificar as coordenadas que a definem de acordo. Por exemplo:
Se a cobra possui 3 segmentos e está na posição [(200,200)(210,200)(220,200)] para ela se mover para a esquerda as coordenadas devem ficar [(190,200),(200,200)(210,200)]. Isso significa que, iniciando da posição mais à direita da cobrinha, cada posição recebe as coordenadas da posição imediatamente anterior e a última posição (como não existe posição anterior a ela) nesse caso se move 10 para a direita (o que significa subtrair 10 da coordenada x).

Dicas para esse passo:

- Para ajudar na leitura do código é bom definir macros que representam as posições (CIMA = 0, DIREITA = 1, BAIXO = 2 ESQUERDA = 3), por exemplo.
- A cobrinha deve começar o jogo com uma direção pré-definida. Como as nossas coordenadas estão crescendo para a direita, isso significa que a “cabeça” (início) da nossa cobra está do lado esquerdo, por isso ela deve começar andando para a esquerda. Isso pode variar de acordo com as coordenadas que definem a sua cobrinha. Mas ela deve começar o jogo andando na direção da sua cabeça.
- Para que a cobrinha se movimente deve-se modificar suas coordenadas de acordo com a direção.
- Para selecionar a direção da cobrinha com o teclado existe um evento no pygame semelhante ao QUIT que é o KEYDOWN. As opções interessantes de teclas para nosso caso estão armazenadas em *event.key* e podem ser: K_UP, K_DOWN, K_LEFT, K_RIGHT.

Passo 4: Adicionar um clock

Ao fazer a cobrinha se movimentar no passo anterior você deve ter percebido que ela se movimenta muito rapidamente. Para consertar esse problema devemos adicionar um clock de forma que a tela só seja redesenhada depois de um certo tempo, fazendo com que a cobrinha se mova mais lentamente.

Dicas para esse passo:

- O pygame tem um objeto time.Clock().
- O objeto time.Clock() possui uma função tick(framerate em milisegundos)

Passo 5: Adicionar a maça

Nesse passo devemos criar a maçã e desenhá-la na tela. Como foi definido que nossos quadrinhos são 10x10 esse também será o tamanho da nossa maçã.

Dicas para esse passo:

- A maçã deve ser desenhada dentro dos limites da tela. Como a nossa tela é 600x600 a última coordenada x e y que a maçã pode obter é 590 (pois ela possui tamanho 10). Criar uma função para obter as coordenadas dentro dos limites é uma boa prática.
- A maçã tem que ser desenhada alinhada com a cobra, a posição (201,201) não é válida para a maçã.
- As funções/objetos do pygame relevantes para esse passo são: Surface, fill e blit.
- A cor vermelha é representada em RGB por (255,0,0).

Passo 6: Comer a maçã

Nesse passo iremos implementar a ação da cobrinha colidir com a maçã para comer a mesma e “crescer” um segmento.

Dicas para esse passo:

- Implementar uma função para detectar se a cabeça da cobra colidiu com a maçã. (No nosso caso a cabeça da nossa cobra é determinada pelas coordenadas da posição [0]).
- Cada vez que a cobra colidir com a maçã ela deve crescer um segmento. (Método *append* da lista python).
- Caso haja colisão uma nova maçã deve aparecer em uma outra posição aleatória da tela.

Passo 7: Adicionar detecção de colisão da cobra com a janela do jogo e com seu corpo

Para esse passo devemos identificar se a cabeça da cobra atingiu os limites da janela (nossas coordenadas vão de 0 a 600 nos eixos x e y) ou se sua cabeça está ocupando a mesma posição de algum outro segmento de seu corpo.

Dicas para esse passo:

- Caso haja colisão o jogo deverá ser terminado e fechado (pygame.quit() e exit()). Futuramente podemos adicionar uma tela de Game Over.

Passo 8: Adicionar um grid

Nesse passo iremos adicionar linhas de grid em nossa janela para marcar os segmentos.

Dicas para esse passo:

- A cada 10 unidades na vertical e na horizontal devemos desenhar uma linha utilizando a função: *pygame.draw.line(Surface, color, start_pos, end_pos)*

Passo 9: Adicionar score

Nesse passo iremos adicionar uma contagem de pontos. Inicialmente o jogador começa com 0 ponto. Cada maçã representa um ponto (ou quantos pontos forem definidos). Para adicionar o score deve-se utilizar o módulo pygame.font, que é um módulo para carregar e renderizar fontes.

Dicas para esse passo:

- Criar uma fonte, por exemplo: *pygame.font.Font('freesansbold.ttf', 18)*.
- Criar um variável para fazer a contagem de pontos que inicia com 0 e é somada a cada vez que a cobra colide com a maçã.
- Utilizar o método *pygame.font.Front.render(text, antialias, color)* para desenhar um texto em uma superfície. Antialias é um bool que define se os caracteres devem ter as bordas suavizadas if true the characters will have smooth edges.
- Para imprimir o score pode-se utilizar um padrão parecido com: *'Score: %s' % (score)*.
- A função get_rect do objeto Font retorna um retângulo que deverá ser posicionado na tela utilizando o método *topleft(x, y)*.

Passo 10: Adicionar game over

Nesse passo iremos adicionar um texto de Game Over para ser utilizado no lugar de simplesmente fechar o jogo quando a cobra colide com as bordas ou consigo mesma.

Dicas para esse passo:

- Utilizar novamente o objeto Font.
- Outro local para a localização do retângulo com o texto pode ser o midtop.
- Criar um loop fora do loop principal do jogo para ficar imprimindo a tela de game over até o evento de quit.
- Funções de ajuda: *pygame.display.update(), pygame.time.wait(500)*