Notas de Projeto

* Comentários sobre o código
* Comentários pessoais
* Agradecimentos

Comentários sobre o código

A primeira parte do codigo são as linhas 1 e 2, nas quais temos o *"import"* de duas bibliotecas:

- random, na qual importamos o modulo randint

- pyxel, a biblioteca base onde será construído todo o jogo.



A partir da linha 5 temos as definições das variaveis globais que serão utilizadas no decorrer do codigo, são elas:

x\_head, y\_head --> Responsáveis por marcar a posição inicial da cabeça da cobra nas coordenadas x e y.

x\_body, y\_body --> Uma lista de [X's] e [Y's] para formar o corpo da cobra, essas listas são baseadas nas coordenas da cabeça da cobra.

apple\_x, apple\_y --> Coordenadas da posição das maçãs que aparecerão durante o jogo, é inicializada na mesma coordenada da cabeça da cobra.

move\_x, move\_y --> Variaveis que farão o movimento da cobra, variam entre {-1, 0, 1}.

up, down, left, right --> Movimento que será feito pelas variaveis move\_x e move\_y.

fps --> O jogo roda a 60 fps

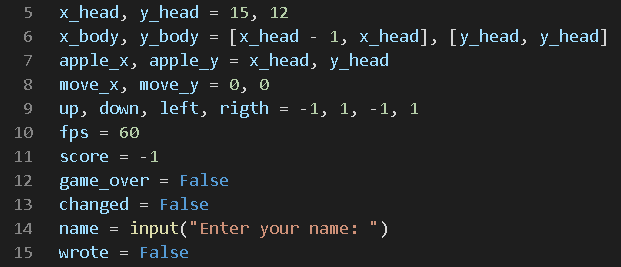
score --> Contagem de maçãs comidas pelo player, é inicializado em -1 pois como inicializamos as coordenadas apple\_x = x\_head e apple\_y = y\_head, devemos fazer esse desconte de 1 ponto.

game\_over --> boolean, inicializado em False.

changed --> boolean, marcador de troca de musicas, inicializado em False.

name --> imput do nome do player, é realizado a cada vez que o programa é inicializado.

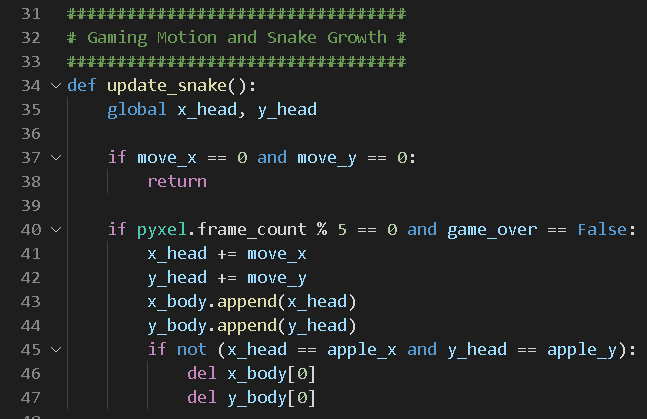
wrote --> boolean, marcador para a escrita do nome do player no leaderboard, inicializado em False.



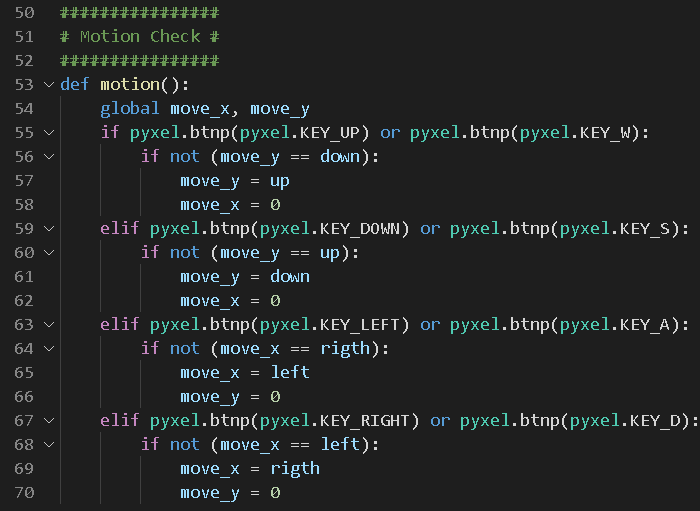
Na linha 23 temos a função write, responsável por abrir o arquivo de texto(.txt) do leaderboard e nele incerir o nome do player junto as seu score, nesse função a variavel wrote é mudada para True, caso contrario seria escrito o leaderboard a cada frame do jogo.



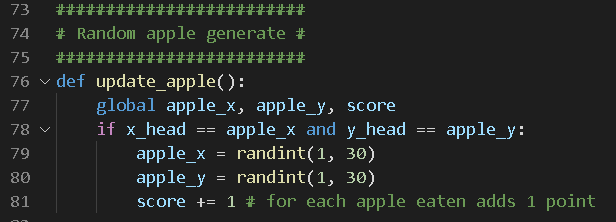
Na linha 34 temos a função update\_snake, responsável por definir o movimento da cobra e o seu crescimento, o movimento é dado pelo append(x\_head e y\_head) nas lista x\_body e y\_body, ao mesmo tempo que deletamos o primeiro termo da lista, assim dando a noção/sensação de movimento para o usuário, caso a cabeça a cobra(ultimos elementos das listas) estejam na mesma coordenada que a maçã o delete não ocorre, dessa forma a cobra cresce uma unidade(1 pixel) a cada maçã comida.



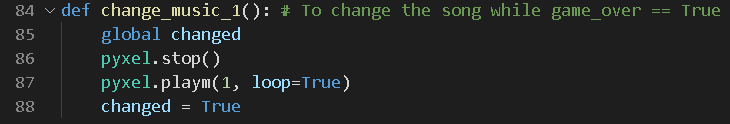
Na linha 53 temos a função motion, responsável por corresponder as variaveis move\_x e move\_x com up, down, left e right, de acordo com a tecla apertada pelo usuário, também impedindo q a cobra atravesse o próprio corpo proibindo movimento opostos ao mesmo tempo.



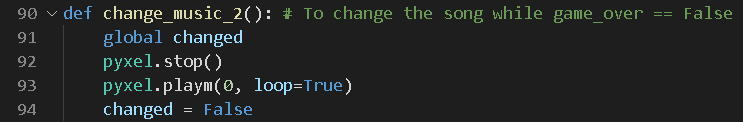
Na linha 76 temos a função update\_apple, responsável por gerar uma nova coordenada, por meio do modulo randint que baixamos anteriormente, para a maçã a cada vez e é comida pela cobra, tmb adicionando um ponto ao score.



Na linha 84 temos a função change\_music\_1, responsável por parar a musica do jogo e começar a musica de game\_over, nessa etapa é alterada a variavel changed para True.

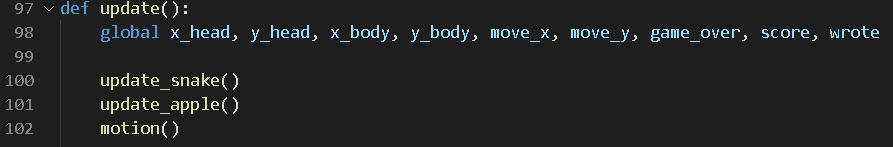


Na linha 90 temos a função change\_music\_2, responsável por para a musica do modo game\_over e começar a musica do jogo, nessa etapa é alterada a variavel changed para False novamente.

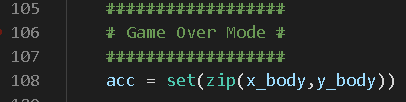


Na linha 97 temos a função update, a função update é uma função standard da biblioteca pyxel a qual é atualizada a cada frame, ou seja todas as funções anteriores que criamos serão chamadas dentro dessa.

Começamos chamando as funções: update\_snake, update\_apple e motion;



Na linha 108 entramos no game\_over mode, declarando a variavel acc(um conjunto das listas x\_body e y\_body) que usaremos como verificação para a colisão da cabeça da cobra com seu próprio corpo.



Na linha 110 verificamos o tamanho do conjunto acc com o tamanho la lista y\_body(como as listas x\_body e y\_body têm o mesmo tamanho usamos apens uma como padrão de medida), desse modo, caso o tamanho do conjunto seja menor que o da lista significa que temos alguma coordenada repetida na lista, assim, mostrando que houve uma colisão com o rabo da cobra, caso a colisão ocorra a variavel game\_over é mudada para True.



Na linha 113 verificamos se as coordenadas da cabeça da cobra(x\_head e y\_head) colidem com as paredes, caso a colisão ocorra a variavel é mudada para True.

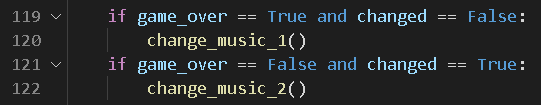


Na linha 116 verificamos se o jogo está no modo game\_over == True e se a variavel wrote ainda não foi mudada para True, caso isso ocorra chamaremos a função write, desse modo evitamos repetições no leaderboard.

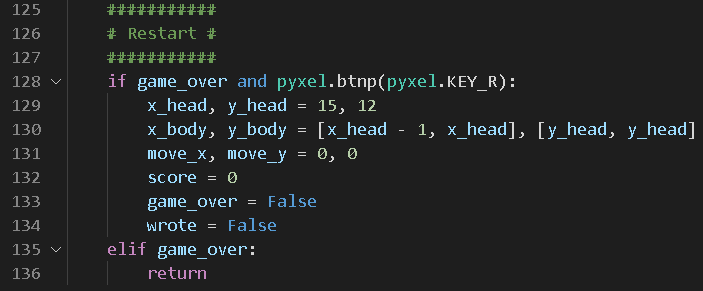


Na linha 119 verificamos se o jogo está no modo game\_over == True e se a variavel changed ainda não foi mudada para True, caso isso ocorra chamaremos a função change\_music\_1.

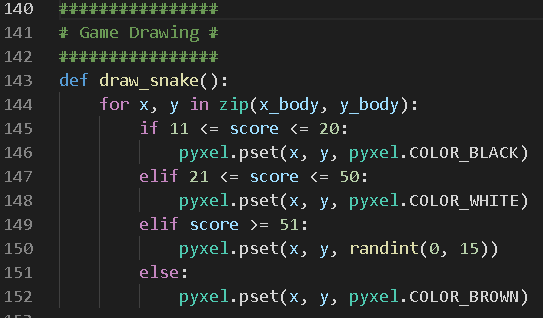
Na linha 121 verificamos se o jogo está no modo game\_over == True e se a variavel changed já foi mudada para Ture, caso isso ocorra chamaremos a função change\_music\_2.



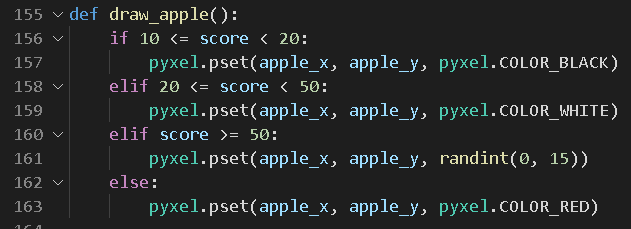
Da linha 128 a 135 temos uma condicional para um possivel restart do jogo, desse modo, restaurando as variaveis globais para seus valores padrões.



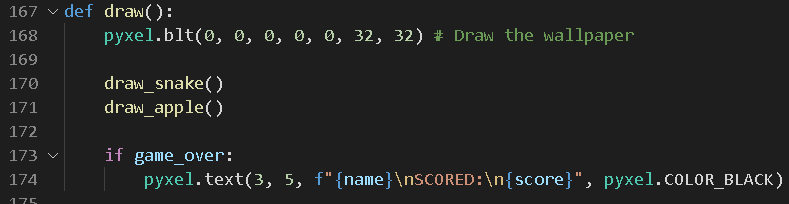
Na linha 143 temos a função draw\_snake, nessa função por meio de um “zip” das listas x\_body e y\_body, usamos o modulo “pset” da biblioteca pyxel para desenhar um pixel em cada coordenada desse “zip”, trazemos também algumas condicionais para que assim tenhamos cores diferentes de acordo com a pontuação do jogador.



Na linha 155 temos a função draw\_apple, aqui temos algumas condicionais que se correspondem as da função anterior, em cada uma delas desenhando a maçã em um pixel da tela usando as coordenadas já obtidas na função update\_apple.



Na linha 167 temos a função draw, assim como a função update, essa é uma função base da biblioteca pyxel, a qual é atualizada a cada frame. Começamos a fulção com o modulo “btl” da biblioteca, o qual desenhará o wallpaper do jogo (presente no arquivo arts\_sets). Logo após, chamamos as funções de desenho anteriores: draw\_snake e draw\_apple.



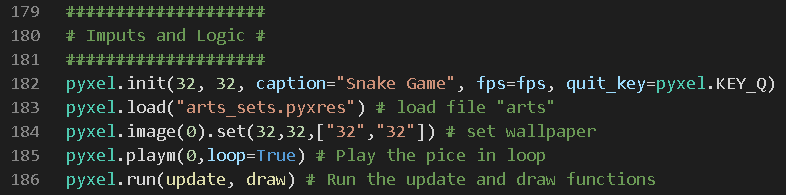
Na linha 182 temos o modulo “init”,assim como as funções updade e draw é um modulo base da bibliteca, que recebe como argumentos as dimenções da tela (32 x 32), o título do jogo, fps ( já mostrado anteriormente) e um argumento de tecla para o fechamento do programa (tecla Q).

Na linha 183 temos o modulo “load” que abre o arquivo arts\_sets, responsável pelas musicas e pelo wallpaper.

Na linha 184 temos o modulo “image” seguido do modulo “set” responsáveis por setar a imagem do wallpaper nas dimenções adequadas da tela.

Na linha 185 temos o modulo “playm” que faz com que a musica do arquivo seja tocada durante a execução do jogo, recebe o argumento “loop=True” para tocar em loop durante toda a execução.

Na linha 186 temos o modulo “run”, assim como modulo “init” também é base da biblioteca, sendo ele responsável por chamar as funções update e draw.



Comentários pessoais

O jogo possui um bug, o qual não tenho certeza de onde vem nem o que ocasiona, mas por a cobra não poder atravesar seu proóprio corpo, as vezes quando apertamos teclas para sentidos perpendicularmente diferentes ela tende a atravezar o próprio corpo. Por exemplo:

A cobra seguindo a direção CIMA e apertamos as teclas ESQUERDA e BAIXO, quase que simultaneamente, a cobra vai tentar ir pra baixo sem virar a esquerda ocasionando no game over.

Imagino que tenha a ver com alguma precariedade dos metodos de reconheicimento de teclas do Pyxel, como btn, btnp… Porém já tentei a troca de todos e não muda em nada, o bug perciste. Desse modo, tentei também criar condicionais muito especificas para proibir esse movimento e mesmo assim o bug perciste. Por isso, considero que esse foi e ainda é o maior desafio do codigo.

Agradecimentos

Faz-se fulcral agradecer ao professor Fábio Macêdo Mendes:

Foi um pazer participar da diciplina de APC, em tua turma, aprendi muito nesse período, desde de o inicio foi uma materia super agradavel de se participar e lidar ao longo do curso, e toda tua ajuda, auxilios e ensinos foram exencial para meu desenvolvimento.

Muito Obrigado!