UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
BACHARELADO EM CIÊNCIAS E TECNOLOGIA

VINICIUS LUTERO CAMILLO
RUTILENO GABRIEL CAMILO DA SILVA

LINGUAGEM DE PROGRAMAÇÃO

NATAL-RN
2022

Introdução

Neste projeto final da matéria de Linguagem de Programação, LiP, do curso de Bacharelado em Ciências e Tecnologia, C&T, começamos com a ideia de criar um jogo baseado na linguagem “Python” e, com o auxílio dos conhecimentos adquiridos com os professores Francisco Vidal e Sérgio Queiroz. Além disso, vale ressaltar a importante ajuda do monitor Lucas, no que diz respeito à solução de problemas que envolvem o código e em algumas dicas essenciais na organização do mesmo.

Nesse cenário, utilizamos a completa e diversificada biblioteca do Pygame, como principal ferramenta para operacionalizar o jogo, além, também, de outras diversas outras bibliotecas que, mais na frente, poderão ser citadas no texto e/ou vistas no próprio código do jogo.

```
from turtle import width
import pygame
from sys import exit
import random
from random import randint,choice
from player import Player
from ObstacleCorrection import Obstacle
from background import Background
from score import Score
```

Imagem 1 - bibliotecas utilizadas no código

Objetivo

Como já foi citado anteriormente no relatório parcial, o objetivo inicial do jogo era criar algo em que muito se assemelhava aos jogos *Super Mario Bros* e o *T-rex game,* respectivamente das empresas Nintendo e Google. Nesse sentido, após algumas quebras de cabeça corrigindo os “bugs” e erros no código, chegamos na conclusão de que ele, o jogo, deveria ser algo mais original e, de certa forma, toma como base os jogos anteriores porém, foram criados e idealizados por nós.

Dessa forma, surgiu o então conhecido “Endless Runner”, um projeto que, à priori, não vai ser totalmente concluído com a entrega final, porém, não significa que o jogo não está pronto, pelo contrário, está completamente funcional. Nessa conjuntura, é de extrema importância algumas melhorias e implementações de novas funcionalidades que, no futuro, darão uma nova vida ao jogo.

Desde o último envio, foram expendidos incansáveis horas de correções e de implementações de funcionalidades e de embelezamentos, sejam eles visuais ou, até mesmo, estruturais no código e dentro do jogo. Nesse cenário, podemos destacar a completa implementação da parte audiovisual do jogo, como efeitos sonoros, música de fundo, background funcional, contagem de pontos e a mais trabalhosa delas, a total transformação do código para classes e métodos com a total limpeza e “fluidez” do código atual.

Implementações estas que serão mostradas a seguir:

Imagem 2 - Background

```python
from ast import increment_lineno
import pygame
import os

class Score():

    def __init__(self,screen):
        self.font = pygame.font.Font('freesansbold.ttf',70)
        self.screen = screen
        self.actual_score = 0
        self.high_score = 0
        #self.score_message = self.font.render('PONTOS',False, 'Black')
        #self.score_message_rect = self.score_message.get_rect(topleft = (600 ,40))


    def show_score(self):

        #self.screen.blit(self.score_message,self.score_message_rect)

        self.score_value = self.font.render(f'{self.actual_score}',False,('Grey'))
        self.score_value_rect = self.score_value.get_rect(topleft = (400,300))
        self.screen.blit(self.score_value,self.score_value_rect)

    def increment_score (self):
        self.actual_score += 1

        if self.actual_score > self.high_score:
            self.actual_score = self.high_score
```

Imagem 3 - Biblioteca Score (autoria própria)

Método

Os métodos utilizados no jogo atual, são provenientes da própria linguagem do python e de suas diversas bibliotecas que dão bastante liberdade para criação e desenvolvimento de várias funcionalidades no que concerne à criação do jogo em si.

Nesse sentido, estamos utilizando bibliotecas como a pygame, principal que faz o jogo funcionar, além de diversas outras, inclusive, algumas delas criadas por nós, como já foi citado anteriormente.

O principal objetivo de clarear o código foi concluído com sucesso, e, em sua nova versão, está de uma forma mais limpa e que proporciona um bom entendimento no que diz respeito à divulgação do mesmo em nossos respectivos Githubs, para, de certa forma, participar do movimento do código-aberto, para ajudar e incentivar novas pessoas a criar seus próprios projetos, sejam eles jogos ou soluções para o dia-a-dia deles

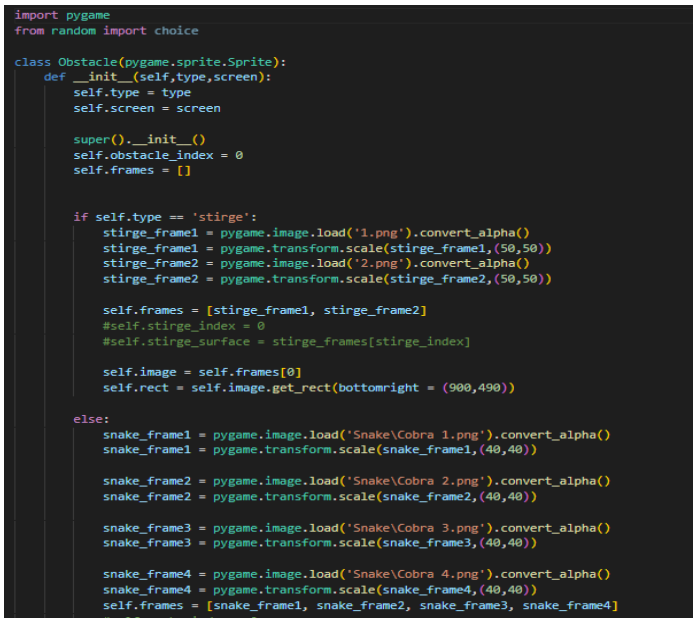
Seguem algumas imagens do código atual do jogo:

```python
class Player(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()
        #Walking
        player_surface1 = pygame.image.load('WalkSoldier\Soldado (1).png').convert_alpha()
        player_surface2 = pygame.image.load('WalkSoldier\Soldado (2).png').convert_alpha()
        player_surface3 = pygame.image.load('WalkSoldier\Soldado (3).png').convert_alpha()
        player_surface4 = pygame.image.load('WalkSoldier\Soldado (4).png').convert_alpha()

        self.player_walk = [player_surface1,player_surface2,player_surface3,player_surface4]
        self.player_index = 0
        #self.player_surface = player_walk[player_index]

        self.image = self.player_walk[self.player_index]
        self.rect = self.image.get_rect(bottomright = (200,550))
        self.gravity = 0

        #Jumping
        player_jump1 = pygame.image.load('jumpSoldier\Tile050.png').convert_alpha
        player_jump2 = pygame.image.load('jumpSoldier\Tile051.png').convert_alpha
        player_jump3 = pygame.image.load('jumpSoldier\Tile052.png').convert_alpha
        player_jump4 = pygame.image.load('jumpSoldier\Tile053.png').convert_alpha

        self.player_jump = [player_jump1,player_jump2,player_jump3,player_jump4]
        self.player_index_jump = 0
        #self.player_surface_jump = player_jump[player_index_jump]

        self.image_jump = self.player_jump[self.player_index_jump]

        self.jump_sound = pygame.mixer.Sound('Jump.wav')
        self.jump_sound.set_volume(0.5)
```

Imagem 4 - Código limpo

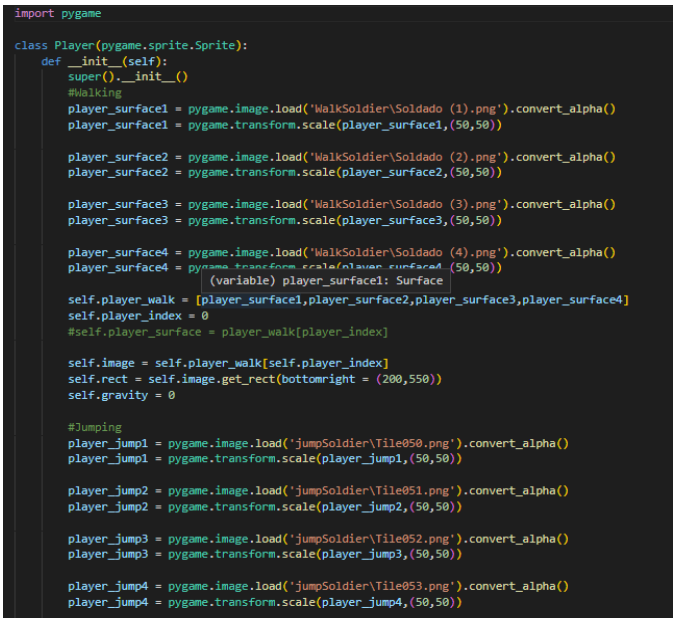
```python
class Obstacle(pygame.sprite.Sprite):
    def __init__(self,type):
        super().__init__()

        if type == 'stirge':
            stirge_frame1 = pygame.image.load('1.png').convert_alpha()
            stirge_frame1 = pygame.transform.scale(stirge_frame1,(50,50))
            stirge_frame2 = pygame.image.load('2.png').convert_alpha()
            stirge_frame2 = pygame.transform.scale(stirge_frame2,(50,50))

            #stirge_index = 0
            self.frames = [stirge_frame1, stirge_frame2]
            #stirge_surface = stirge_frames[stirge_index]
            y_pos = 515
        else:
            snake_frame1 = pygame.image.load('Snake\Cobra 1.png').convert_alpha()
            snake_frame1 = pygame.transform.scale(snake_frame1,(40,40))

            snake_frame2 = pygame.image.load('Snake\Cobra 2.png').convert_alpha()
            snake_frame2 = pygame.transform.scale(snake_frame2,(40,40))

            snake_frame3 = pygame.image.load('Snake\Cobra 3.png').convert_alpha()
            snake_frame3 = pygame.transform.scale(snake_frame3,(40,40))

            snake_frame4 = pygame.image.load('Snake\Cobra 4.png').convert_alpha()
            snake_frame4 = pygame.transform.scale(snake_frame4,(40,40))

            #snake_index = 0
            self.frames = [snake_frame1, snake_frame2, snake_frame3, snake_frame4]
            #snake_surface = snake_frames[snake_index]
            y_pos = 550
```

Imagem 5 - Código Limpo



```python
import pygame
from random import choice

class Obstacle(pygame.sprite.Sprite):
    def __init__(self,type,screen):
        self.type = type
        self.screen = screen

        super().__init__()
        self.obstacle_index = 0
        self.frames = []


        if self.type == 'stirge':
            stirge_frame1 = pygame.image.load('1.png').convert_alpha()
            stirge_frame1 = pygame.transform.scale(stirge_frame1,(50,50))
            stirge_frame2 = pygame.image.load('2.png').convert_alpha()
            stirge_frame2 = pygame.transform.scale(stirge_frame2,(50,50))

            self.frames = [stirge_frame1, stirge_frame2]
            #self.stirge_index = 0
            #self.stirge_surface = stirge_frames[stirge_index]

            self.image = self.frames[0]
            self.rect = self.image.get_rect(bottomright = (900,490))

        else:
            snake_frame1 = pygame.image.load('Snake\Cobra 1.png').convert_alpha()
            snake_frame1 = pygame.transform.scale(snake_frame1,(40,40))

            snake_frame2 = pygame.image.load('Snake\Cobra 2.png').convert_alpha()
            snake_frame2 = pygame.transform.scale(snake_frame2,(40,40))

            snake_frame3 = pygame.image.load('Snake\Cobra 3.png').convert_alpha()
            snake_frame3 = pygame.transform.scale(snake_frame3,(40,40))

            snake_frame4 = pygame.image.load('Snake\Cobra 4.png').convert_alpha()
            snake_frame4 = pygame.transform.scale(snake_frame4,(40,40))
            self.frames = [snake_frame1, snake_frame2, snake_frame3, snake_frame4]
```

Imagem 6 - Código Limpo



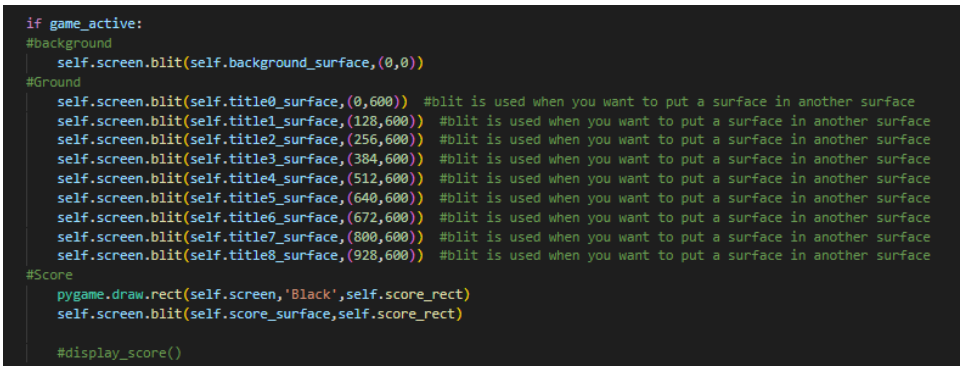
```python
import pygame

class Player(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()
        #Walking
        player_surface1 = pygame.image.load('WalkSoldier\Soldado (1).png').convert_alpha()
        player_surface1 = pygame.transform.scale(player_surface1,(50,50))

        player_surface2 = pygame.image.load('WalkSoldier\Soldado (2).png').convert_alpha()
        player_surface2 = pygame.transform.scale(player_surface2,(50,50))

        player_surface3 = pygame.image.load('WalkSoldier\Soldado (3).png').convert_alpha()
        player_surface3 = pygame.transform.scale(player_surface3,(50,50))

        player_surface4 = pygame.image.load('WalkSoldier\Soldado (4).png').convert_alpha()
        player_surface4 = pygame.transform.scale(player_surface4,(50,50))
                          (variable) player_surface1: Surface
        self.player_walk = [player_surface1,player_surface2,player_surface3,player_surface4]
        self.player_index = 0
        #self.player_surface = player_walk[player_index]

        self.image = self.player_walk[self.player_index]
        self.rect = self.image.get_rect(bottomright = (200,550))
        self.gravity = 0

        #Jumping
        player_jump1 = pygame.image.load('jumpSoldier\Tile050.png').convert_alpha()
        player_jump1 = pygame.transform.scale(player_jump1,(50,50))

        player_jump2 = pygame.image.load('jumpSoldier\Tile051.png').convert_alpha()
        player_jump2 = pygame.transform.scale(player_jump2,(50,50))

        player_jump3 = pygame.image.load('jumpSoldier\Tile052.png').convert_alpha()
        player_jump3 = pygame.transform.scale(player_jump3,(50,50))

        player_jump4 = pygame.image.load('jumpSoldier\Tile053.png').convert_alpha()
        player_jump4 = pygame.transform.scale(player_jump4,(50,50))
```

Imagem 7 - Código Limpo

Conclusão

A experiência adquirida com a criação desse jogo, é de extrema importância para nossas carreiras e futuras profissões, tanto no que concerne o uso e a aplicação da linguagem em si, mas também no trabalho em equipe e na criação e solução de problemas em diversas situações no dia-a-dia

Além disso, vale ressaltar que nosso outro objetivo também é facilitar o entendimento de terceiros sobre nosso código, pois, facilitará bastante a participação e divulgação do jogo, fomentando a cultura do código aberto.

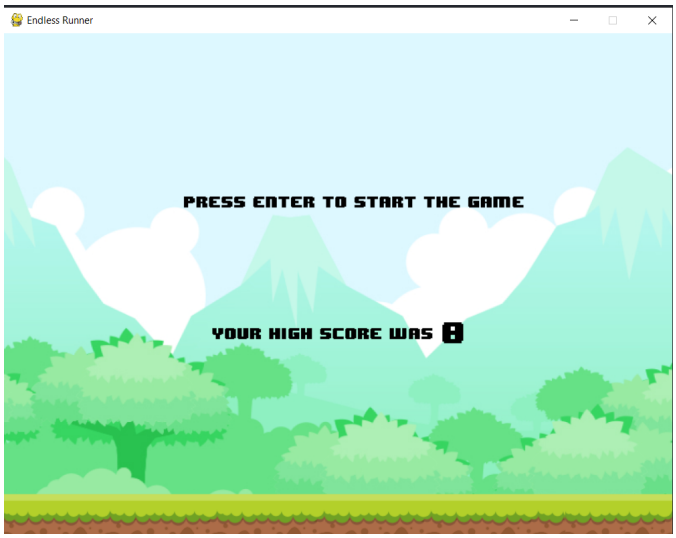


```python
if game_active:
#background
    self.screen.blit(self.background_surface,(0,0))
#Ground
    self.screen.blit(self.title0_surface,(0,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title1_surface,(128,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title2_surface,(256,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title3_surface,(384,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title4_surface,(512,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title5_surface,(640,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title6_surface,(672,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title7_surface,(800,600))  #blit is used when you want to put a surface in another surface
    self.screen.blit(self.title8_surface,(928,600))  #blit is used when you want to put a surface in another surface
#Score
    pygame.draw.rect(self.screen,'Black',self.score_rect)
    self.screen.blit(self.score_surface,self.score_rect)

    #display_score()
```

Imagem 8 - código comentado

```python
class EndlessRunner:
    """Overall class with screen and important variables"""

    def __init__(self):
        """Initialize the game and create game resources"""
        pygame.init()

        pygame.display.set_caption('Jogo.png') #setting the name of the game

        self.screen = pygame.display.set_mode((1000,750)) #widht,height #(()) its a tuple
        self.clock = pygame.time.Clock()
        self.test_font = pygame.font.Font('pixeltype.ttf',30) #font type and font size
        self.game_active = False
        self.start_time = 0

    def display_score(self):
        """It not actual an score, it is a timer"""
        current_time = pygame.time.get_ticks() - self.start_time
        score_surface = self.test_font.render(f'{current_time}',False,(64,64,64)) #curren_time NEED TO BE AN INT
        score_rect = score_surface.get_rect(topleft = (425,50))
        self.screen.blit(score_surface,score_rect)
        #print(current_time)
```

Imagem 9 - código comentado



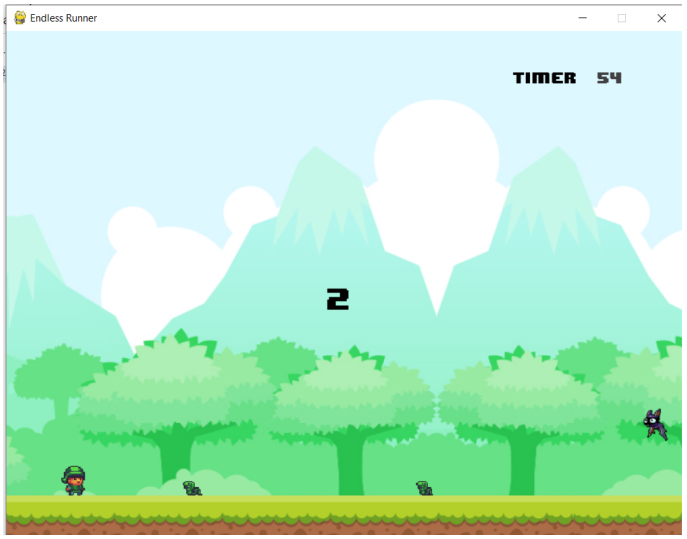


Imagem 11 - Menu do jogo

Imagem 12 - Jogando

Como executar o Jogo

→ Dentro do arquivo .ZIP, tem algumas das pastas que estamos utilizando, no momento, com os arquivos do jogo em si.

→ O arquivo executável do jogo, no momento, é o “EndlessRunner”, arquivo Python.

→ A princípio, onde ele está é só descompactar, abrir o arquivo diretamente da pasta e executá-lo, sem nenhum problema.

→ Recomendo somente executar o jogo, sem debugar o código, se for no Virtual Studio Code, só apertar o play no canto superior direito.

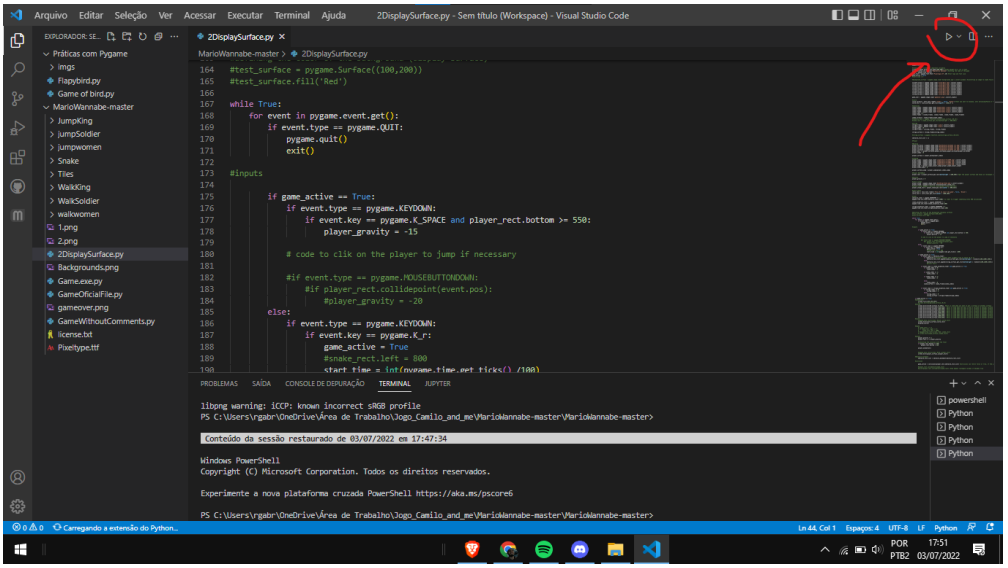


Imagem 10 - Como iniciar o jogo