**Projeto Final**

Aluno: Bruna Lima Farias

**Quis Star Wars**

**Propósito:**

O projeto é uma combinação de um sistema de cadastro de planetas, personagens e eventos do universo Star Wars, um sistema de perguntas e respostas que testa o conhecimento dos jogadores sobre esses planetas e um mapa em 2d da galáxia do universo dos filmes

**Funcionamento:**

* **Cadastro de Planetas, Personagens e Eventos:**

O usuário pode cadastrar informações sobre planetas do universo Star Wars, incluindo o nome do planeta, personagens que nasceram nele e eventos relacionados a esse planeta

Isso é feito usando a opção "1. Cadastrar Planeta" no menu.

Os planetas cadastrados são armazenados em um dicionário chamado cadastro\_planetas, neste já possuindo dados pré cadastrados

* **Cadastro de Perguntas sobre Planetas:**

Os usuários podem cadastrar perguntas relacionadas a um planeta específico. Cada pergunta deve incluir a pergunta, as opções de respostas, a resposta correta e uma explicação

Isso é feito usando a opção "2. Cadastrar Pergunta para Planeta" no menu

As perguntas são associadas a um planeta específico e armazenadas nos dados do planeta no dicionário cadastrar\_planeta\_pergunta

* **Lista Planetas:**

Exibe uma lista de planetas que estão cadastrados quando selecionado a opção "3. Lista Planetas" no menu

* **Quiz:**

O jogador pode iniciar o quiz relacionado a um planeta específico, respondendo a perguntas sobre esse planeta, selecionando a opção "4. Iniciar Quiz"

Os pontos são calculados com base nas respostas corretas e registrados para o jogador

Os resultados do quiz são armazenados no dicionário cadastro\_planetas com o nome do jogador

* **Galáxia 2D**

A opção "5. Mostrar Galaxia 2D" no menu uma representação simplificada da galáxia do universo de Star Wars

* **Pontuação**

Depois de terminar de cadastrar planetas, perguntas e realizar o quiz, o jogador pode ver sua pontuação final

A opção "6. Encerrar" no menu encerra o programa e exibe as pontuações finais de todos os jogadores listados

**Evidências:**

* O código contém funções para validar entradas não vazias, cadastrar planetas, cadastrar perguntas

Texto

Descrição gerada automaticamente

* O quiz exibe perguntas, opções resposta e uma explicação sobre as respostas

Texto

Descrição gerada automaticamente

**Testes:**

**Cadastro de Planetas:**

Adicionado da maneira correta

Texto

Descrição gerada automaticamente

Input (“ ”) sem nada dá como entrada invalida

Texto

Descrição gerada automaticamente

**Cadastro de Perguntas:**

Adicionado da maneira correta

Texto

Descrição gerada automaticamente

**Quiz:**

Mostrado da maneira correta, com as perguntas e validando a resposta

Texto

Descrição gerada automaticamente

**Lista de Planetas:**

É mostrada a lista com os planetas cadastrados

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

**Mostrar Galáxia 2D:**

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Fundo preto com letras brancas

Descrição gerada automaticamente com confiança média

**Encerramento:**

Ao encerrar é mostrada a pontuação de acordo com o jogador e o planeta

Uma imagem contendo Diagrama

Descrição gerada automaticamente

**Códigos:**

projeto\_star\_wars.py

*"""*

*Cadastro sobre filmes de Star Wars com um quiz sobre os filmes I II III IV V VI*

*"""*

*# Sistema de Cadastro de Planetas*

from galaxia\_2d import Galaxia2DApp

*# dicionario base*

cadastro\_planetas = {

    "Tatooine": {

        "personagens": "Anakin Skywalker, Luke Skywalker",

        "eventos": "Pod race, Duelo com Darth Vader",

        "perguntas": [

            {

                "pergunta": "Qual é o planeta natal de Anakin Skywalker?",

                "respostas": ["a) Tatooine", "b) Coruscant", "c) Naboo"],

                "resposta\_correta": "a",

                "explicacao": "Anakin Skywalker nasceu em Tatooine."

            }

        ]

    },

    "Endor": {

        "personagens": "Leia Organa",

        "eventos": "Batalha de Endor",

        "perguntas": [

            {

                "pergunta": "Onde os Ewoks vivem?",

                "respostas": ["a) Alderaan", "b) Coruscant", "c) Endor"],

                "resposta\_correta": "c",

                "explicacao": "Os Ewoks vivem no planeta Endor."

            }

        ]

    }

}

*# dicionário separado para as pontuações*

pontuacoes = {}

*# tratamento de erro, verifica e entrada nao esta vazia*

def entrada\_nao\_vazia(mensagem):

    while True:

        entrada = input(mensagem).strip()

        if entrada:

            return entrada

        else:

            print("Por favor, insira uma entrada valida")

def cadastrar\_planeta():

    nome\_planeta = entrada\_nao\_vazia("Digite o nome do planeta: ")

    personagens = entrada\_nao\_vazia("Digite os personagens que nasceram neste planeta: ")

    eventos = entrada\_nao\_vazia("Digite os eventos relacionados a este planeta: ")

*# adiciona no dicionario*

    cadastro\_planetas[nome\_planeta] = {

        "personagens": personagens,

        "eventos": eventos,

        "perguntas": []

    }

def cadastrar\_pergunta\_planeta():

    nome\_planeta = entrada\_nao\_vazia("Digite o nome do planeta para cadastrar a pergunta: ")

*# verificar se planeta existe no cadastro*

    if nome\_planeta in cadastro\_planetas:

        pergunta = entrada\_nao\_vazia("Digite a pergunta: ")

        respostas = [input("Digite a alternativa (a): "), input("Digite a alternativa (b): "), input("Digite a alternativa (c): ")]

        resposta\_correta = entrada\_nao\_vazia("Digite a resposta correta (a, b, c): ")

        explicacao = entrada\_nao\_vazia("Digite uma explicação da resposta: ")

*# adiciona no dicionario*

        cadastro\_planetas[nome\_planeta]["perguntas"].append({

            "pergunta": pergunta,

            "respostas": respostas,

            "resposta\_correta": resposta\_correta,

            "explicacao": explicacao

        })

    else:

        print(f"O planeta {nome\_planeta} não está cadastrado. Cadastre primeiro.")

*# Funcao para mostrar lista de planetas que estao no dicionario de cadastro*

def lista\_planetas():

    print("\nLista de Planetas:")

    for planeta in cadastro\_planetas.keys():

        print(f"\n- {planeta}")

*# retorna perguntas sobre um planeta especifico*

def get\_pergunta(nome\_planeta):

    return cadastro\_planetas.get(nome\_planeta, {}).get("perguntas", [])

def executar\_quiz():

    nome\_planeta = input("Digite o nome do planeta para o quiz: ")

*# verifica se o planeta existe no cadastro*

    if nome\_planeta in cadastro\_planetas:

        jogador = input("Digite o nome do jogador: ")

*# usando as perguntas cadastradas para cada planeta específico*

        pontuacao = 0

        perguntas = cadastro\_planetas.get(nome\_planeta, {}).get("perguntas", [])

        if not perguntas:

            print(f"Não há perguntas cadastradas para o planeta {nome\_planeta}.")

            return

        print(f"Quiz sobre o planeta {nome\_planeta} para {jogador}:\n")

*# dicionário sobre uma pergunta especifica relacionada a um planeta*

        for pergunta\_data in perguntas:

            print(pergunta\_data["pergunta"])

            for resposta in pergunta\_data["respostas"]:

                print(resposta)

            resposta = input("Sua resposta (a, b ou c): ")

            if resposta == pergunta\_data["resposta\_correta"]:

                print("Correto! Você ganhou um ponto.")

                pontuacao += 1

            else:

                print("Errado. A resposta correta é:", pergunta\_data["resposta\_correta"])

                print(pergunta\_data["explicacao"])

        if jogador not in pontuacoes:

            pontuacoes[jogador] = {}

        pontuacoes[jogador][nome\_planeta] = pontuacao

    else:

        print(f"O planeta {nome\_planeta} não está cadastrado. Cadastre-o primeiro.")

def executa\_estrela\_da\_morte():

    print("  .          \_\_---\_\_")

    print(".     .   .-'...:...'-.               .          .")

    print("         / .  . : .\_\_ .\                                         .")

    print("  .     /........./  \ .\  .   .                            .")

    print("       / :  :   :| () | :\                  .        .")

    print("      :...........\\_\_/....:         .")

    print("      |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|              .                     .")

    print("      |...................|               .")

    print(".     :  :  :   :   :   : :                          .            .")

    print("    .  \................./      .            .")

    print("        \  .  . : .  .  /   .                                .")

    print("     .   \.\_.........\_./  .        .                   .")

    print("            -..\_\_\_..-                .         .")

    print("")

def main():

    while True:

        print("\n")

        print("                    8888888888  888    88888")

        print("                   88     88   88 88   88  88")

        print("                    8888  88  88   88  88888")

        print("                       88 88 888888888 88   88")

        print("                88888888  88 88     88 88    888888")

        print(" ")

        print("                88  88  88   888    88888    888888")

        print("                88  88  88  88 88   88  88  88")

        print("                88 8888 88 88   88  88888    8888")

        print("                 888  888 888888888 88   88     88")

        print("                  88  88  88     88 88    8888888")

        print("\nMenu:")

        print("1. Cadastrar Planeta")

        print("2. Cadastrar Pergunta")

        print("3. Lista de Planetas")

        print("4. Iniciar Quiz")

        print("5. Mostrar Galaxia 2D")

        print("6. Encerrar")

        opcao = input("Escolha uma opção: ")

        if opcao == "1":

            cadastrar\_planeta()

        elif opcao == "2":

            cadastrar\_pergunta\_planeta()

        elif opcao == "3":

            lista\_planetas()

        elif opcao == "4":

            executar\_quiz()

        elif opcao == "5":

            galaxia\_app = Galaxia2DApp()

            galaxia\_app.run()

        elif opcao == "6":

            break

        else:

            print("Opção inválida")

    print("Pontuações finais:")

    for jogador, dados in pontuacoes.items():

        for planeta, pontuacao in dados.items():

            print(f"{jogador} - Planeta {planeta}: Pontuação: {pontuacao}")

    print(executa\_estrela\_da\_morte())

if \_\_name\_\_ == "\_\_main\_\_":

    main()

galaxia\_2d.py

*# galaxia\_2d.py*

*# import do kivy para a interface grafica*

from kivy.app import App

from kivy.uix.widget import Widget

from kivy.graphics import Ellipse, Color

from kivy.clock import Clock

from math import cos, sin, radians

*# tela para visualização do modelo 2D*

class Galaxia2DWidget(Widget):

    def \_\_init\_\_(self, \*\*kwargs):

        super(Galaxia2DWidget, self).\_\_init\_\_(\*\*kwargs)

*# define os planetas, velocidade, distancia, raio e cor*

        self.planets = {

            'Tatooine': {'radius': 5, 'distance': 30, 'color': (1, 0.8, 0), 'angle': 5, 'orbit\_speed': 10},

            'Coruscant': {'radius': 25, 'distance': 120, 'color': (0, 0.5, 1), 'angle': 0, 'orbit\_speed': 6},

            'Endor': {'radius': 18, 'distance': 180, 'color': (0, 0.8, 0), 'angle': 0, 'orbit\_speed': 8},

            'Hoth': {'radius': 22, 'distance': 250, 'color': (1, 1, 1), 'angle': 0, 'orbit\_speed': 7},

        }

        for planet, info in self.planets.items():

            with self.canvas:

                Color(\*info['color'])

                info['ellipse'] = Ellipse(pos=self.calculate\_position(planet), size=(info['radius'] \* 2, info['radius'] \* 2))

        Clock.schedule\_interval(self.update, 1 / 60.0)

*# posição do planeta*

    def calculate\_position(self, planet):

        info = self.planets[planet]

        angle\_rad = radians(info['angle'])

        x = self.width / 2 + info['distance'] \* cos(angle\_rad)

        y = self.height / 2 + info['distance'] \* sin(angle\_rad)

        return (x - info['radius'], y - info['radius'])

*# animação dos planetas, atualizando a posição da elipse*

    def update(self, dt):

        for planet, info in self.planets.items():

            info['angle'] += dt \* info['orbit\_speed']  *# ajuste velocidade do movimento orbital*

            info['ellipse'].pos = self.calculate\_position(planet)

*# chama a tela para abrir o aplicativo de visualização*

class Galaxia2DApp(App):

    def build(self):

        return Galaxia2DWidget()

if \_\_name\_\_ == "\_\_main\_\_":

    galaxia\_app = Galaxia2DApp()

    galaxia\_app.run()