NOME: Matheus Felipe Ruiz Gonçalves **RA:** 80268-2021
NOME: Ademir Santos Damião **RA:** 74650-2021
NOME: Luis Otavio Oliveira Marques **RA:** 708503-2021

ORGANIZAÇÃO DAS CLASSES NO CÓDIGO

Inicialmente, foi feito o mapeamento das classes necessárias no projeto que ficaram como separadas em arquivos próprios .py, mas apenas para que fosse visualizável as classes como um todo que foram utilizadas sem ficar uma embaixo da outra em apenas um arquivo python.

CLASSE GOL

Na classe “Gol” foram utilizados os atributos “jogador” para conseguirmos identificar qual foi o jogador que fez o gol. Nesse caso, optamos por não definir em jogador a classe “Jogador” pois em nenhuma das consultas que envolvem ambas as classes é necessário saber mais informações sobre o jogador, logo deixamos em classe separadas que são utilizadas para consultas específicas.

CLASSE JOGADOR

Nessa classe foram adicionados os atributos “nome” e “gols” pois possuímos uma funcionalidade que busca consultar os gols de determinado jogador.

CLASSE TIME

A classe “Time” foi criada para estarmos podendo sempre acessar informações do time que principalmente são demandadas quando são carregadas as partidas na função carregar_partidas( ) no qual precisa do nome do time o código dele porque essas propriedades auxiliam no processo de consulta das partidas a partir do atributo “codigo”.

CLASSE PARTIDA

A classe em geral somente foi criada para que fosse possível armazenar as informações de forma organizada e foi criado os atributos “visitante” e “mandante” como sendo do tipo da classe Time devido a consulta por seleção no qual utiliza o atributo “codigo”, mas caso contrário não seria necessário armazenamos como sendo uma classe ao invés de apenas do nome do mandante e visitante.

ARQUIVO MAIN.PY

FUNÇÕES

criar_partida(grupo)

A função serve para criar uma nova partida no qual recebe um dicionário no qual é possível acessar as chaves para conseguir criar o objeto Partida e retorná-lo para que chama a função.

carregar_grupo(grupo)

A função serve para carregar todas as partidas com base no grupo do tipo str que é passado como parâmetro da função. No Início, caso as partidas não estejam já inseridas na lista PARTIDAS[ ] ele já insere para que não ocorra nenhum problema ao buscar um determinado grupo e depois percorre nessa lista de partidas de dicionários com as informações da partida onde possuem a chave “group” e depois vai criando instâncias da classe Partida e adicionando na lista que vai ser retornada pela função com uma lista dos jogos que aconteceram no grupo.

consultar_por_grupo(grupo)

A função inicialmente chama um outro método chamado carregar_grupo(grupo: str) que instancia as partidas com base no nome do grupo informado como parâmetro e depois retorna essa mesma lista com as partidas. Posteriormente, é chamada a função com_partidas( ) para verificar se essa lista possui partidas e, caso tenha partidas, ele irá executar o método informacoes_partida( ) para todas as partidas presente na lista que basicamente tem o propósito de mostrar as informações da partida na tela.

carregar_rodada(rodada)

A operação inicialmente busca a lista de dicionários com as informações das partidas com base no número da rodada digitada pelo usuário que possui uma interface prévia explicando qual numeração digitar para a rodada desejada. Depois, com base nisso ele cria uma lista com as partidas da rodada , depois adiciona as partidas da rodada na lista e posteriormente retorna essa lista de partidas da rodada.

carregar_partidas(rodada)

O procedimento basicamente percorre em um loop FOR por todas as rodadas e depois pelas suas respectivas partidas para posteriormente ir criando as partidas com a função criar_partida( ) e adiciona todas as partidas da copa do mundo para a lista PARTIDAS[ ] para que todas as partidas estejam disponíveis para ser acessadas por outros métodos.

carregar_gols(jogador)

A função busca percorrer por todas as rodadas, depois por todas as partidas dessas rodadas e posteriormente ele pergunta se há as chave ‘goals1’ ou ‘goals2’ na partida atual do laço de repetição. Caso haja uma dessas chaves, ele vai juntar todas as chaves da lista de ‘goals2’ em ‘goals1’ para depois instanciar objetos do tipo Gol e adicionar a lista que armazena todos os gols da copa do mundo (GOLS [ ])

partidas_carregadas( )

Verifica se na lista PARTIDAS [ ] que contém todas as partidas do copa do mundo está com as instâncias já presente na lista. Caso não esteja, então ele chama o método carregar_partidas( ) que insere todas as partidas na lista.

com_partidas(partidas)

Basicamente é verificado se a lista informada como argumento na função tem alguma conteúdo dentro e, caso possua, é passado por todos os elementos da lista usando a função map ( ) que por sua vez vai executar o método informacoes_partida ( ) em todas as partidas presente na lista para ao final retorna verdadeiro indicando que foi achado partidas dentro da lista, senão irá ser retornado falso para indicar que não há partidas dentro dessa lista.

consultar_por_selecao(codigo_selecao)

Inicialmente é inserido as partidas da copa do mundo na lista PARTIDAS [ ], caso elas não tenham sido previamente inseridas. Depois é criado uma lista com as partidas que possuem o código de seleção informado como parâmetro no time mandante ou visitante.
Posteriormente, ele somente imprime as informações das partidas, caso haja partidas com essa seleção no método com_partidas( ).

consultar_por_estadio(estadio)

Inicialmente é inserido as partidas da copa do mundo na lista PARTIDAS [ ] ,caso elas não tenham sido previamente inseridas. Depois é criado uma lista com as partidas que possuem o nome do estádio informado como argumento da função.
Posteriormente, ele somente imprime as informações das partidas, caso haja partidas nesse estádio informado no método com_partidas( ).

consultar_por_cidade(cidade)

Inicialmente é inserido as partidas da copa do mundo na lista PARTIDAS [ ], caso elas não tenham sido previamente inseridas. Depois é criado uma lista com as partidas que possuem o nome da cidade informada como argumento da função.
Posteriormente, ele somente imprime as informações das partidas, caso haja partidas nessa cidade informada no método com_partidas( ).

consultar_gols_jogador(jogador)

Primeiramente é verificado se a lista de gols de todas as partidas (GOLS [ ]) já foi previamente preenchida e, caso não haja nenhum conteúdo na lista, é chamado o método carregar_gols( ) que instancia e insere na lista GOLS[ ]. Posteriormente, é criado uma lista de gols do jogador que é resultado da iteração da lista de gols onde será filtrado somente os gols que possuem o nome do jogador informado como argumento na função.
Caso haja gols desse jogador, então é mostrado o nome dele novamente e a quantidade de gols marcados na copa do mundo.

consulta_rodadas(rodada)

A função acaba inicialmente carregando as partidas do número da rodada que é passado como parâmetro na função no qual possui previamente uma interface mostrando a qual fase se refere cada numeração. Posteriormente, é mostrada a quantidade de partidas que ocorreram nessa rodada com base no tamanho da lista de partidas da rodada e depois é chamado o método com_partidas ( ) que informa detalhadamente as informações das partidas que ocorreram na rodada.

consultar_por_grupo(grupo)

No início é inserido em uma lista as partidas do nome do grupo que é informado como argumento da função que vem de retorno da função carregar_grupo ( ) no qual retorna uma lista com instâncias da classe Partida que possuem o grupo informado. Posteriormente, é chamado o método com_partidas ( ) passando a lista de partidas do grupo como argumento para estar imprimindo as informações das partidas, se houver alguma partida com o grupo informado.

get_qtde_vitorias_mandante( )

A função inicialmente declara uma variável para armazenar a quantidade de vitórias por parte do time mandante, depois é feito uma iteração na lista de todas as partidas no qual a cada partida em que a quantidade de gols do mandante é maior do que a quantidade de gols do visitante para então ser incrementado a variável com a quantidade de vitórias do time mandante que posteriormente é retornada pela função

get_qtde_vitorias_visitante( )

A função inicialmente declara uma variável para armazenar a quantidade de vitórias por parte do time visitante, depois é feito uma iteração na lista de todas as partidas no qual a cada partida em que a quantidade de gols do visitante é maior do que a quantidade de gols do mandante para então ser incrementado a variável com a quantidade de vitórias do time visitante que posteriormente é retornada pela função

get_qtde_empates( )

A função inicialmente declara uma variável para armazenar a quantidade de empates, depois é feito uma iteração na lista de todas as partidas no qual para cada partida em que a quantidade de gols do visitante é igual a quantidade de gols do mandante é incrementado a variável com a quantidade de empates que posteriormente é retornada pela função

analistar_resultados( )

Consiste em um método que para conferir os dados estatísticos de vitórias e empates da copa do mundo no qual carrega as partidas na lista de todas as partidas (PARTIDAS [ ]), caso não foram previamente carregadas e posteriormente é informado na tela do usuário as informações de vitórias do mandante e visitante, além do número de empates no qual é chamado as funções get_qtde_vitorias_mandante( ), get_qtde_vitorias_visitante ( ), get_qtde_empates ( )