Copa+

Luis Felipe Cunha

Departamento de ciências exatas - Universidade Estadual de Feira de Santana (UEFS)

CEP 44036-900 – Feira de Santana – BA – Brasil

luis.lipecunha@gmail.com

Abstract. *Copa+ is a registration application for all information related to the groups, teams and first phase matches of the 2022 FIFA World Cup. of each game. At the end of all registrations, statistics from the first phase and the clashes formed in the round of 16 will be made available for viewing.*

Resumo. *O Copa+ é um aplicativo de cadastro para todas as infromações relacionadas aos grupos, equipes e jogos da primeira fase da Copa do Mundo FIFA 2022. Informações relacionadas ao dia, horário e local de cada confronto entre as equipes serão fornecidas pelo usuário, além do resultado de cada jogo. Ao final de todos os cadastros, estatisticas da primeira fase e os confrontos formados nas oitavas de final serão disponibilizados para visualização.*

1. Introdução

A Copa do Mundo FIFA, maior evento futebolístico do mundo ocorre de quatro em quatro anos levando entretenimento aos quatro cantos do planeta. O Torneio surgiu em 1930, para mudar as perspectivas mundiais com relação ao esporte e quebrar as barreiras sociais existentes no planeta. [LANCE,2021].

Um aplicativo de cadastro para todas as informações do torneio foi solicitado aos alunos do curso de Engenharia de Computação da Universidade Estadual de Feira de Santana (UEFS). O aplicativo, nomeado como Copa+ é desenvolvido com a linguagem de programação Python. A aplicação devera permitir o cadastro dos grupos com suas seleções, além dos resultados e informações relacionadas ao dia, horário e local dos confrontos.

Todas as informações serão salvas em arquivo, e todos os dados devem ser manipulados sem o comprometimento das informações fornecidas pelo usuário. Ao final um relatório poderá ser solicitado pelo usuário, nele, diversas informações estatísticas serão informadas, como as médias de gol por grupo e geral, além de todas as informações relacionadas ao confronto em que uma seleção marcou mais gols.

As seleções classificadas para as oitavas de final e os confrontos formados a partir desta classificação também serão informados pela aplicação caso o usuário solicite a informação.

2. Metodologia

Para um melhor desenvolvimento do programa sessões tutoriais do componente MI algoritmos foram realizadas para compartilhamento de ideias e fatos ao longo do desenvolvimento da aplicação. A utilização de arquivos do tipo JSON para uma melhor manipulação dos dados foi compartilhada ao grupo, assim como, a segmentação das

informações em diferentes arquivos, facilitando a busca pelas informações cadastradas.

Ao longo do cadastro das informações, tudo deve ser validado para que a integridade do arquivo não seja comprometida, para este fim, listas de verificação foram criadas, controlando os inputs do usuário.

Buscando uma melhor organização do arquivo, dicionários foram criados para armazenamento das informações, utilizando como chave o grupo e o nome das duas equipes participantes de um confronto. Para armazenar todas as informações inseridas pelo usuário, quatro arquivos JSON foram gerados, um para os grupos e suas seleções, outro para os confrontos e os seus respectivos dia, horário e local, seguindo a mesma forma, um arquivo com os confrontos e seus resultados foi gerado e as tabelas preenchidas com os pontos, gols marcados e saldo de gols se localizam em um outro arquivo.

2.1. Requisitos e funcionalidades

O aplicativo deve permitir que o usuário cadastre todas as equipes e seleções participantes do Torneio, uma verificação deve ser feita para impedir que uma seleção já inserida anteriormente seja cadastrada novamente, além de não permitir o cadastro de um mesmo grupo mais de uma vez.

Os confrontos devem ser gerados automaticamente a partir das seleções cadastradas, é ofertado que o usuário possa inserir o dia, horário e local de cada confronto criado, além do resultado de cada um desses jogos.

As opções de cadastro, exclusão ou edição de uma informação devem estar disponíveis, além disso, verificações precisam ser feitas para evitar que as modificações feitas pelo usuário não comprometam a base de dados da aplicação. O usuário poderá ter a liberdade de pausar o cadastro no momento que desejar, e caso isso ocorra, as informações cadastradas anteriormente não devem ser perdidas.

Com o preenchimento de todos os campos relacionados as informações gerais dos jogos, os classificados para as oitavas de final, assim como os confrontos gerados a parir dessa classificação devem ser informados caso o usuário queira. É de escolha do usuário a exibição das médias de gol por grupo e geral, além da partida em que uma seleção marcou mais gols e todas as informações relacionadas ao confronto em que isto ocorreu.

2.2 Algoritmo

Inicialmente para utilização dos arquivos a biblioteca JSON com todas as suas funções é importada. Classes são definidas para alocação das informações em memória no momento de cadastro das informações, uma é responsável por armazenar o grupo e as seleções e a outra o dia, horário e local de um confronto.

Uma única função, responsável pelo preenchimento dos quatro arquivos utilizados é criada, ela recebe como parâmetro o arquivo que será preenchido e o dicionário que será alocado neste arquivo. O método seek (0) é utilizado para evitar erros caso o usuário tenha interrompido o preenchimento das informações e queira retornar em seguida com a continuação do cadastro.

Outra função realiza o processo de leitura do arquivo, o recebendo como parâmetro e retornando todos os dados que estavam armazenados nele.

O menu principal funciona a partir de um While True que encerra o programa caso

a opção sair seja escolhida pelo usuário. Sete opções estão disponíveis para escolha, são elas: Acesso ao menu dos grupos, acesso ao menu dos jogos, acesso ao menu dos resultados, visualização da tabela de informações de um grupo, contendo os pontos, gols marcados e o saldo de gols para cada seleção em ordem decrescente, estatísticas da primeira fase e confrontos das oitavas de final.

Todas as variáveis, listas e dicionários que serão utilizados ao longo do programa são declarados, além da lista oficial com os oito grupos da copa de A à G, para fins de verificação, evitando que o usuário crie grupos inexistentes no torneio.

Caso a opção escolhida seja a primeira, o programa direciona o usuário para um outro menu, este por sua vez totalmente relacionado aos grupos e suas equipes. A função responsável por carregar os dados de um arquivo, recebe o arquivo dos grupos como parâmetro e retorna todas as informações relacionadas a ele. Duas listas, são geradas a partir desses dados, uma contendo todos os grupos previamente cadastrados no arquivo e outra com todas as seleções presentes no arquivo.

As funções de cadastrar, editar ou excluir um grupo são apresentadas neste novo menu, a função de cadastro só é disponível após a contagem de grupos inseridos na lista de grupos, caso o somatório alcance o valor 9 (oito grupos mais um coringa, previamente inserido no arquivo), o cadastro é bloqueado e o usuário é informado. A função len do Python é utilizada para realizar o somatório e duas condicionais informam ao usuário a situação do cadastro.

Em uma situação em que a lista de grupos ainda não atingiu seu valor máximo, o cadastro é liberado. Um While com as opções deseja continuar, sim ou não, é colocado para que o usuário sai do cadastro quando quiser. A classe equipes, responsável pelo grupo e as seleções é declarada, assim como, à tupla que armazena as quatro seleções.

O print com todos os grupos carregados do arquivo é feito, assim como o print dos grupos que foram cadastrados enquanto o usuário continuou no while de cadastro. Os inputs para escolha dos grupos e das quatro seleções participantes do mesmo são colocados à disposição, todos possuem verificações que evitam a repetição de um cadastro já feito, como mostra a figura 1 abaixo.

```
e.grupo = input("Grupo: ").upper().strip()
while e.grupo in lista_grupos or e.grupo in lista_grupos_parcial or e.grupo not in grupos_oficiais:
    e.grupo = input("\nGRUPO INVALIDO: Qual grupo deseja cadastrar =>").upper().strip()
lista_grupos_parcial.append(e.grupo)


e.t1 = input("\nPRIMEIRA SELECAO: ").upper().strip()
while e.t1 in lista_selecoes or e.t1 in lista_selecoes_parcial:
    e.t1 = input("\nSELECAO ESCOLHIDA JA FOI CADASTRADA: Qual selecao deseja cadastrar =>").upper().strip()
lista_selecoes_parcial.append(e.t1)
```

Figura 1. Cadastro dos grupos e seleções

A figura 1 acima detalha o processo exemplificando a escolha do grupo e da primeira seleção participante a ser cadastrada, um while no input do grupo não permite que o usuário escolha um grupo que esteja presente no arquivo, ou que foi cadastrado

anteriormente por ele naquele mesmo instante, assim como, em caso do grupo não estar listado entre os grupos oficiais da FIFA.

Para as seleções não é permitido o cadastro de uma equipe presente no arquivo e que já foi cadastrada por ele naquele instante. As funções. upper () e. strip (), são utilizadas para respectivamente, deixar todos os nomes em tamanho maiúsculo e apagar espações residuais em branco deixados pelo usuário antes ou depois de um nome.

Após o cadastro do grupo com suas quatro seleções, os dicionários que armazenaram estas informações no arquivo são construídos. Primeiro para o arquivo dos grupos, uma tupla declarada anteriormente é carregada com as quatro seleções, logo em seguida essa tupla é colocada como valor de um dicionário que possui como chave o grupo que corresponde as seleções cadastradas.

Para o arquivo das tabelas uma lista contendo um dicionário é criada. O dicionário possui como chave cada seleção cadastrada e valor uma lista preenchida com os pontos, gols marcados e saldo de gols de cada equipe, neste momento todos os valores são nulos, já que nenhum resultado foi inserido no programa. Ao final a lista é declarada como valor de um dicionário que possui com o chave o grupo relacionado as quatro seleções.

A imagem 2 abaixo, exibe o processo de criação dos dois dicionários citados acima.

```
#GRUPOS
tupla = (e.t1,e.t2,e.t3,e.t4)
agrupamento[e.grupo] = tupla


#TABELA
lista_tabela = [{e.t1:[None,None,None],e.t2:[None,None,None],e.t3:[None,None,None],e.t4:[None,None,None]}]
tabelao[e.grupo] = lista_tabela
```

Figura 2. Criação dos dicionários de grupos e tabelas

Os confrontos são gerados de forma automática após o cadastro das quatro seleções. A disposição é feita de forma manual em dois dicionários, uma para os confrontos com os dias, horários e locais de cada um e outro para o resultado de cada confronto.

Para o dicionário das informações gerais do confronto a chave é montada com os dois times participantes do confronto e valor uma lista de dicionários que possuem como chave o dia, horário e local, o valor de cada uma dessas informações é nulo, já que nenhum cadastro com relação a esse quesito foi liberado ao usuário neste momento do programa.

Para o arquivo dos resultados o chaveamento é feito da mesma forma, um dicionário possui como chave o confronto e valor uma lista contendo um dicionário que armazena o resultado de cada equipe do confronto.

A figura 3 abaixo exemplifica o processo de criação dos dois dicionários citados.

Figura 3. Criação dos dicionários com as informações gerais dos confrontos e seus resultados

Após o processo de criação dos dois dicionários, cada um é carregado em uma lista que se torna o valor de um dicionário que possui como chave o grupo que acabou de ser cadastrado pelo usuário.

Ao final de cada cadastro feito, a soma da quantidade de elementos existentes na lista grupos que foi carregada no arquivo, com a lista grupos carregada enquanto o usuário cadastrou os grupos é feito, em caso de soma igual a nove, um break é acionado e o programa informa que todos os grupos foram cadastrados.

Em uma nova situação em que a opção de cadastro seja escolhida, uma verificação é feita para descobrir se ainda existem grupos pendentes para cadastro, caso todos estejam preenchidos, o usuário é informado que o menu escolhido está indisponível.

A segunda opção do menu de cadastro criada par editar seleções em seus grupos está apenas disponível caso exista pelo menos um grupo cadastrado, uma verificação é feita para este fim e o usuário é avisado em uma situação de inexistência de grupos para edição.

Acessando o menu de edição, a lista com todos os grupos cadastrados e exibida e a função escolha grupo é chamada para verificar se o grupo que o usuário escolhera para edição está disponível na lista grupos. A função recebe como parâmetro a lista de grupos e um while mantem o usuário no input caso seja digitado um grupo fora da lista, ao final do processo a função retorna o grupo escolhido.

Após o processo de escolha do grupo o usuário escolhe o time que será inserido, um outro while de verificação não permite que o usuário escolha um time que já está presente na lista de seleções.

Para escolha do time que será removido do grupo, uma lista contendo todas as equipes presentes no grupo escolhido é gerada a partir da função verifica time edição. Ela possui como parâmetro o grupo selecionado para edição e os dados do arquivo dos grupos. No input um while é utilizado para evitar que o usuário queira retirar uma seleção que não está presente na lista de equipes do grupo para edição. Ao final de cada input a lista de seleções geral, remove a seleção excluída e insere a equipe que entrará na base de dados.

A função editar grupo é chamada, ela recebe como parâmetros o grupo, a seleção que irá entrar no grupo e a que será retirada, a modificação deve ser feita nos quatro arquivos utilizados pelo programa e para este fim, somente esta função é utilizada.

O primeiro arquivo a ser editado é o dos grupos, o respectivo arquivo é aberto e carregado, um for é utilizado para localizar o grupo que terá suas seleções editadas, após este processo um outro for percorre as seleções do grupo, ao encontrar a seleção que será retida, uma condicional if a troca pela seleção que será inserida, uma nova lista carrega os valores alterados e o grupo que foi editado recebe esta lista atualizada como novo valor, após o fim do processo o arquivo é recarregado com as mudanças.

A edição do arquivo dos confrontos e resultados é feita de forma diferente, já que nestes, as seleções são chaves de dicionários. A imagem 4 abaixo detalha o processo de edição no arquivo dos confrontos.

```
with open('Confrontos.json','r') as df:
        dados = json.load(df)
    data = {}
    for itens in dados:
        for chave in itens:
            if chave == grupo:
                for confronto in itens[chave]:
                    for times in confronto:
                        if selecao_sair in times:
                            x = times.replace(selecao_sair,selecao_entrar)
                            data[x] = confronto[times]
    for itens in dados:
        for chave in itens:
            if chave == grupo:
                for confronto in itens[chave]:
                    for times in confronto:
                        if selecao_sair not in times:
                            data[times] = confronto[times]
    lista = [data]
    for itens in dados:
        for chave in itens:
            if chave == grupo:
                itens[chave] = lista
    with open('Confrontos.json','w') as df:
        json. dump(dados, df)
```

Figura 4. Edição de seleção no arquivo dos confrontos

Em um primeiro momento o arquivo é aberto e suas informações são carregadas. Um dicionário provisório é declarado, ele será utilizado para armazenar os novos valores modificados.

Uma estrutura de for’s é utilizada para percorrer os dados do arquivo e localizar a chave que será alterada. Ao encontrar as chaves que possuem a seleção escolhida para sair do grupo, a função ponto replace é utilizada para trocar esta seleção com aquela que

será inserida em seu local, o dicionário provisório carrega estas novas chaves e seus respectivos valores.

Em seguida, é necessário buscar as chaves que não foram alteradas para carrega-las no dicionário provisório e atualizar o arquivo da forma correta, o processo é feito da mesma forma, entretanto, ao invés de carregar as chaves em que a seleção que foi excluída esta presente, são carregadas aquelas que não foram alteradas no primeiro processo.

Ao final para que o arquivo retorne no mesmo formato que foi enviado o dicionário é colocado em uma lista e esta é atribuída como valor ao grupo que foi editado, por fim o arquivo é recarregado com as mudanças.

Para o arquivo dos resultados o mesmo processo é feito, por ambos terem o mesmo formato de disposição para os dados. O arquivo das tabelas, possui processo semelhante, entretanto, para este não é necessário realizar o replace na chave, apenas a troca de nomes já é suficiente para edição. Um dicionário provisório é utilizado para armazenar as informações soa longo da edição e este é inserido em uma lista que se torna valor do grupo escolhido para edição.

O terceiro e último menu dos grupos é a exclusão do grupo, este de forma semelhante ao de edição está somente disponível com o cadastro de pelo menos um grupo, uma condicional realiza este processo de verificação. Em caso de grupo disponível para exclusão a função escolha grupo é chamada para que o usuário selecione o grupo que será removido do arquivo, após o processo de escolha a função exclui grupo é chamada para remover este grupo e todas as informações relacionadas a ele em todos os arquivos.

A função recebe como parâmetro apenas o grupo que será removido. Para realizar esta ação a função reescreve as informações de cada arquivo, com exceção do grupo que foi enviado para exclusão, sendo assim novas informações são enviadas ao arquivo de forma atualizada, excluindo o grupo que foi escolhido pelo usuário. O processo é exibido na imagem 5 abaixo.

```
with open('Confrontos.json','r') as df:
    dados = json.load(df)
alteracao_confrontos = {}

for itens in dados:
    for chave in itens:
        if chave != grupo:
            alteracao_confrontos[chave] = itens[chave]
dados = [alteracao_confrontos]

with open('Confrontos.json','w') as df:
    json. dump(dados, df)
```

Figura 5. Processo de exclusão de um grupo no arquivo dos confrontos

Exemplificando o processo exibido na figura 5 com o arquivo dos confrontos, um dicionário provisório é declarado, após isso uma estrutura de for's escreve nesse dicionário todos os grupos e suas informações com exceção grupo escolhido para

exclusão, o dicionário é atribuído a uma lista e está é carregada no arquivo para sua atualização.

Fora do while do menu de cadastro estão localizadas as funções que preenchem os arquivos após o cadastro, todas as informações cadastradas são alocadas em memória pela classe, os dicionários são colocados como parâmetro da função preencher arquivo que os atualiza, caso exista uma informação preexistente neles. Para evitar erros esse preenchimento é feito apenas se o dicionário dos grupos estiver previamente preenchido com alguma informação, evitando que dicionários vazios sejam inseridos no arquivo e comprometam a integridade dos dados.

Seguindo para o menu dos jogos, este só estará disponível com o cadastro de pelo menos um grupo, quatro opções estão disponíveis para o usuário, a visualização das informações de todos os confrontos de um grupo, com os seus dias, horários e locais, o cadastro do dia, horário e local de todos os confrontos de um grupo e a exclusão das informações de um confronto ou de todos os confrontos de um grupo. Após a escolha da opção desejada, os dados dos arquivos dos confrontos são carregados para futuras verificações.

Para visualizar os confrontos de um grupo e suas informações o usuário deve escolher o grupo, respeitando a lista de grupos disponíveis no momento. Com o grupo definido a função print confronto é chamada, recebendo como parâmetro o grupo escolhido e os dados dos confrontos. Uma estrutura de for's busca o grupo no arquivo e ao encontra-lo exibe todos os confrontos atribuídos a ele.

Caso a opção selecionada seja a de cadastro uma soma é feita para verificar a quantidade de cadastros previamente preenchidos, o que evita que o usuário acesse esse menu com todos as informações preenchidas. A função calculo verificação confronto recebe como parâmetro cada grupo oficial da Copa do Mundo 2022 e os dados dos confrontos, o processo de cálculo é mostrado na figura 6 abaixo.

```
soma = 0
for itens in df:
    for chave in itens:
        if chave == grupo:
            for confrontos in itens[chave]:
                for chapa in confrontos:
                    for atributos in confrontos[chapa]:
                        if atributos['DIA'] != None and atributos['HORARIO'] != None
                        and atributos['LOCAL'] != None:
                            soma+=1
return soma
```

Figura 6. Calculo de quantos cadastros foram realizados

Uma estrutura de for's encontra o grupo enviado como parâmetro e ao encontra-lo verifica se todas as informações relacionadas aos dias, horários e locais de seus confrontos estão diferentes de "NONE", ou nulo, caso esteja, soma-se um, atestando que o cadastro está feito, a soma final é retornada pela função.

Ao final da soma de cada grupo, todos os valores enviados são somados, sendo este menor que 48 (Número total de jogos) o acesso ao menu de cadastro é permitido, em caso de bloqueio o usuário será avisado.

O usuário irá escolher qual grupo terá as informações de seus confrontos cadastradas, a função escolha grupo é chamada e logo em seguida a função cadastra grupo confronto entra recebendo como parâmetro o grupo escolhido e os dados dos confrontos. Para realizar o cadastro do dia horário e local dos confrontos. Dento da função caso o grupo escolhido já tenha o cadastro dos seus confrontos preenchidos o usuário será avisado, em caso contrário as informações começaram a ser cadastradas.

Cada confronto é exibido e o seu dia, horário e local é informado pelo usuário, os seis confrontos do grupo serão cadastrados de uma vez, os inputs são armazenados em uma classe denominada informações partidas, que é declarada no começo do processo como mostra a imagem 7 abaixo.

```
for itens in info:
    for chave in itens:
        if chave == grupo:
            for confrontos in itens[chave]:
                for chapa in confrontos:
                    for atributos in confrontos[chapa]:
                        i = Informacoes_partida
                        if atributos['DIA'] == None and atributos['HORARIO'] == None
                            and atributos['LOCAL'] == None:
                            print("VOCE IRA CADASTRAR O DIA - LOCAL - HORARIO DAS
                                PARTIDAS =>\n")
                            print('\nVOCE IRA CADASTRAR O CONFRONTO A SEGUIR \n =>
                                {} <=\n'.format(chapa))
                            i.dia = input('DIGITE A DATA DESSE CONFRONTO: \n').upper
                                ().strip()
                            i.local = input("\nDIGITE O LOCAL DESSE CONFRONTO: \n"
                                ).upper().strip()
                            i.horario = input("\nDIGITE O HORARIO DESSE CONFRONTO:
                                \n").upper().strip()
                            atributos['DIA'] = i.dia
                            atributos['HORARIO'] = i.horario
                            atributos['LOCAL'] = i.local
```

Figura 7. Cadastro do dia, horário e local de um confronto

Uma estrutura de for's busca o grupo escolhido pelo usuário e ao encontra-lo exibe cada confronto e pergunta em seguida o dia, horário e local daquele confronto, ao final as chaves correspondentes a cada item recebem a informação digitada pelo usuário. O arquivo é atualizado após a finalização do cadastro.

O usuário também pode escolher editar uma informação de um confronto, para isso ele escolherá o grupo e as duas seleções que fazem parte do confronto, no momento de escolha das seleções uma lista de verificação é gerada com as equipes participantes do grupo escolhido e um while evita que o usuário escolha seleções de fora dessa lista.

A edição só é permitida caso os confrontos do grupo tenham suas informações previamente preenchidas, para isso a função verifica preenchimento retorna True em caso de cadastro completo e informa ao usuário caso o cadastro esteja pendente. Em uma situação de permissão o print dos confrontos é feito e o usuário realiza a edição que desejar.

Após escolher o grupo e as seleções o usuário escolhe qual informação deseja alterar, uma cadeia de condicionais leva o usuário para modificar a informação a partir do atributo escolhido para edição. Após isso a nova informação que será colocada no local é informada pelo usuário e a função realiza edição do confronto é chamada. A figura 8 abaixo exibe o processo de edição.

```
for itens in info:
        for chave in itens:
            if chave == grupo:
                for confrontos in itens[chave]:
                    for chapa in confrontos:
                        if se1 in chapa and se2 in chapa:
                            for atributos in confrontos[chapa]:
                                atributos[modificador] = nova_info
    with open('Confrontos.json','w') as dados_confrontos:
        json.dump(info,dados_confrontos)
```

Figura 8. Alteração da informação de um confronto

A função recebe como parâmetros as duas seleções que fazem parte do confronto a ser editado, o grupo ao qual este confronto pertence, os dados do arquivo dos confrontos, o item que será modificado (dia, horário ou local) e a nova informação que será alocada a este item.

Uma estrutura de for's encontra o grupo e logo em seguida as duas seleções do confronto, após isso definem o novo valor do atributo escolhido para mudança, ao final o arquivo é atualizado com as informações.

O usuário pode também excluir os dados de um confronto, ou de todos os confrontos de um grupo, para isso, estas duas opções são ofertadas e a partir de sua escolha ele decidira qual grupo sofrerá esta modificação.

Escolhendo excluir as informações de todos os confrontos de um grupo, a função verifica preenchimento é chamada para averiguar se este grupo possui cadastro para excluir, caso não possua uma condicional avisa ao usuário a situação encontrada. Em caso de permissão, ou seja, cadastro completo, um print de todos os confrontos com suas informações preenchidas é feito, logo em seguida a função excluir dados confronto é chamada, recebendo com parâmetro o grupo e os dados dos confrontos.

Para realizar a exclusão dos dados uma cadeia de for's encontra o grupo selecionado e transforma os dias, horários e locais de seus confrontos em "NONE", ou seja, apaga-se tudo que foi preenchido. Ao final o arquivo é devidamente atualizado. O print dos confrontos é feito após a chamada da função para atestar que todas as informações foram apagadas.

Caso o usuário escolha apagar as informações de um confronto apenas, ele além de escolher o grupo, irá informar as duas seleções que fazem parte do confronto escolhido, uma lista de verificação é gerada para evitar a escolha de seleções inválidas e ao final do processo de escolha a função exclui dados de um confronto é chamada, recebendo como parâmetro as duas seleções, o grupo e os dados dos confrontos.

O processo feito é semelhante ao explicado anteriormente, entretanto, além de ir em busca do grupo informado, a cadeia de for's encontra o confronto em que as duas seleções estão inseridas, após isso apaga-se o dia, horário e local do confronto. Ao final o arquivo é carregado com as atualizações. O print dos confrontos é feito antes e depois do processo de exclusão para comprovar ao usuário que os dados foram apagados.

A opção três do menu principal leva o usuário para as informações relacionadas aos resultados dos confrontos. Os dados dos quatro arquivos são carregados com a função leitura dos arquivos e a soma de verificação é feita para atestar que os 48 confrontos estão com os seus respectivos dias, horários e locais preenchidos, caso exista um confronto com cadastro pendente o usuário será informado através de uma condicional que a entrada no menu dos resultados depende deste cadastro.

Em caso de entrada permitida, ele terá um menu de quatro opções, são elas; cadastrar um resultado, excluir resultado, visualizar resultados de um grupo ou sair. As listas contendo os grupos e seleções são carregadas para futuras verificações.

Escolhendo a opção de cadastro, ele decidirá qual grupo terá um de seus resultados preenchidos, a função escolha grupo é chamada, logo em seguida, uma lista de verificação contendo todas as seleções do grupo escolhido é criada para verificar a integridade da escolha das duas seleções que fazem parte do confronto que terá o seu resultado informado.

Esta ação só é permitida depois que a função verifica resultado grupo for chamada, ela atesta se realmente existem resultados para cadastro no grupo escolhido, em caso de afirmação uma nova função chamada verifica resultado atesta que o confronto entre as duas seleções escolhidas está com o seu resultado vago, em caso de retorno True a função cadastra resultado é chamada.

Recebendo como parâmetros o grupo escolhido, as duas seleções participantes do confronto e os dados referentes aos arquivos dos resultados e das tabelas a função cadastra resultado permite ao usuário a inserção do número de gols em um confronto da Copa do Mundo. A figura 9 abaixo exibe parte do processo feito no interior da função.

```
for itens in dados_resultados:
    for chave in itens:
        if chave == grupo:
            for info in itens[chave]:
                for chapa in info:
                    if se1  in chapa and se2 in chapa:
                        for atributos in info[chapa]:
                            x = chapa.split()
                            gols1 = verifica_gols(x[2])
                            gols2 = verifica_gols(x[6])
                            atributos["RESULTADO TIME 01"] = gols1
                            atributos["RESULTADO TIME 02"] = gols2
```

Figura 9. Cadastro do resultado

A função trabalha com uma estrutura de for's que encontra o grupo escolhido e o

confronto em que estão as duas seleções informadas pelo usuário. O Split é colocado na chave do confronto e os índices onde estão alocadas as seleções são colocados como parâmetro da função verifica gols, ela é responsável por evitar que o usuário digite uma letra em um menu do tipo int.

Após retornar o número de gols de cada seleção a função atribui este valor ao resultado de cada time (x [2], primeiro time da chapa e x [6], segundo time). Logo em seguida, na própria função, o somatório de pontos, saldo de gols e gols marcados é feito.

O arquivo de tabelas é carregado, neste arquivo as seleções são a chave de um dicionário que armazenam uma lista, formada pelos pontos (índice 0), gols marcados (índice 1) e saldo de gols (índice 2). Para entrega dos pontos três condicionais são feitas para verificar qual das duas equipes marcou mais gols e a esta são entregues três pontos e se houve empate, um ponto é atribuído para cada.

Para cadastro dos gols marcados, soma-se o valor existente na lista ao informado pelo usuário no momento de cadastro e para o saldo de gols, é feita a diferença de gols entre as duas equipes e após este processo os valores são alocados na lista.

A figura 10 abaixo exemplifica o processo mencionado com o saldo de gols.

```
if info[x[2]][2] == None:
    info[x[2]][2] = gols1 - gols2
elif info[x[2]][2] != None:
    info[x[2]][2] += gols1 - gols2

if info[x[6]][2] == None:
        info[x[6]][2] = gols2 - gols1
elif info[x[6]][2] != None:
    info[x[6]][2] += gols2 - gols1
```

Figura 10. Calculo do saldo de gols de uma seleção

No primeiro cadastro os valores carregados pelo arquivo são nulos, para atestar a integridade dos dados no momento da criação do arquivo, por este motivo, uma condicional é colocada para verificar se o valor está nulo ou não, este processo é feito para as três situações de processamento explicadas acima.

O usuário pode, se assim quiser, excluir o resultado previamente cadastrado, para este fim ele primeiro decidirá o grupo e a partir dessa escolha uma lista de verificação é gerada para que no momento da escolha das duas seleções que terão o resultado do seu confronto excluído, não existam incoerências.

Após o processo de decisão a função verifica preenchimento é chamada para atestar a existência de um resultado para excluir entre as duas equipes, em caso de negação o usuário é informado e em caso contrário a função exclui resultado é chamada.

Recebendo como parâmetro o grupo que terá um de seus resultados excluídos, as duas seleções desse confronto e os dados referentes aos resultados e a tabela, a função exclui resultado realiza o processo contrário da função de cadastro dos resultados, anulando os valores atribuídos a cada time e retirando desses valores os pontos, gols marcados e saldo do arquivo das tabelas, a imagem 11 abaixo exibe o processo mostrando

a retirada de pontos das equipes.

```
if atributos['RESULTADO TIME 01'] > atributos['RESULTADO TIME 02']:
    info[x[2]][0] -= 3

elif atributos['RESULTADO TIME 01'] == atributos['RESULTADO TIME 02']:
    info[x[2]][0] -= 1
    info[x[6]][0] -= 1

elif atributos['RESULTADO TIME 01'] < atributos['RESULTADO TIME 02']:
    info[x[6]][0] -= 3
```

Figura 11. Retirada dos pontos de uma equipe

Ao final do processo o resultado de cada equipe é anulado e os dois arquivos são atualizados com as mudanças feitas.

A visualização do resultado de todos os confrontos de um grupo também é ofertada, para este fim o usuário apenas precisa informar o grupo para exibição a partir da função escolha grupo e logo em seguida a função print resultado é chamada para exibir todas as informações relacionadas aos resultados daquele grupo.

A opção quatro do menu principal permite que o usuário visualiza as tabelas de cada grupo, ou seja, é permitida a visualização dos times pela classificação dos pontos, gols marcados e saldo de gols. Para acessar este menu todos os resultados precisam estar cadastrados, para este fim o cálculo de verificação é feito para permitir a entrada do usuário no menu, em uma situação de cadastro faltante, o usuário será informado.

Para exibir a tabela, a função print tabela grupo é chamada, recebendo como parâmetros o grupo escolhido, os dados referentes aos arquivos dos resultados e das tabelas e a opção escolhida pelo usuário no menu principal, quatro neste caso.

No interior da função, um dicionário para armazenamento dos pontos, gols marcados e saldo de gols de cada equipe é criado, uma estrutura de for's é utilizada para realizar esta distribuição, como mostra a figura 12 abaixo.

```
dici_selecao = {}
for itens in dados_tabela:
    for chave in itens:
        if chave == grupo:
            for info in itens[chave]:
                for selecao in info:
                    dici_selecao[selecao] = {"PONTOS": info[selecao][0],"SALDO DE GOLS":info[selecao][2],"GOLS MARCADOS":info[selecao][1]}
```

Figura 12. Criação do dicionário com as estatísticas das equipes

Para cada chave são atribuídos os índices da lista anteriormente definida, as três

chaves que representam cada atributo são colocadas em uma lista alocada como valor de um dicionário chaveado pela seleção proprietária daquelas informações.

Uma expressão lambda é utilizada para ordenar os dicionários de forma decrescente por cada item, as seleções já ordenadas são colocadas em uma lista, como mostra a figura 13 abaixo.

```
ordenado_pontos = sorted(dici_selecao, key=lambda selecao: dici_selecao[selecao]["PONTOS"],reverse=True)
ordenado_saldo = sorted(dici_selecao, key=lambda selecao: dici_selecao[selecao]["SALDO DE GOLS"],reverse=True)
ordenado_gols = sorted(dici_selecao, key=lambda selecao: dici_selecao[selecao]["GOLS MARCADOS"],reverse=True)
```

Figura 13. Ordenação das equipes por pontos, saldo e gos marcados

Uma condicional é colocada na função para verificar se a opção do usuário foi a de visualização da tabela, em caso positivo o print é feito, ordenando as seleções por colocação em cada item.

Caso a opção escolhida não tenha sido a de visualização da tabela a função retorna a lista ordenada das seleções por pontos, para definição dos classificados e formação das oitavas de final

A opção cinco do menu principal leva o usuário para as estatísticas da primeira fase, os dados de todos os arquivos e as listas de verificação são carregadas, a entrada neste menu depende do total cadastro de todos os resultados, para isso um cálculo somatório é feito para averiguar se todos os 48 confrontos estão com seus respectivos resultados preenchidos, em caso negativo o usuário é informado.

As médias de gols por grupo e geral são informadas neste menu, para realizar este cálculo a função somar gols é criada, ela recebe como parâmetro o grupo e os dados do arquivo das tabelas. Para realizar o cálculo, uma cadeia de for's entra nos dados do arquivo e realiza a soma de todos os gols marcados (índice 1 da lista alocada para cada time), ao final a função retorna o somatório de gols. O cálculo relacionado as médias do grupo e geral é feita no próprio print como mostra a figura 14 abaixo, o somatório de gols de cada grupo é dividido pela quantidade de confrontos e para o geral soma-se a quantidade de gols de todos os grupos e divide-se pelo total de confrontos.

```
print("\nSEGUE ABAIXO AS MEDIAS DE GOL EM CADA GRUPO ==>")
print("\n>{:.2f} GOLS NO GRUPO A".format(sa/6))
print("\n>{:.2f} GOLS NO GRUPO B".format(sb/6))
print("\n>{:.2f} GOLS NO GRUPO C".format(sc/6))
print("\n>{:.2f} GOLS NO GRUPO D".format(sd/6))
print("\n>{:.2f} GOLS NO GRUPO E".format(se/6))
print("\n>{:.2f} GOLS NO GRUPO F".format(sf/6))
print("\n>{:.2f} GOLS NO GRUPO G".format(sg/6))
print("\n>{:.2f} GOLS NO GRUPO H".format(sh/6))
print("\nMEDIA DE GOLS GERAL ==> {:.2f} GOLS".format((sa+sb+sc+sd+se+sf+sg+sh) / 48))
```

Figura 14. Cálculo das médias de gol por grupo e geral

Nesta mesma opção de estatísticas o jogo em que uma equipe fez mais gols e todas as informações relacionadas a este confronto devem ser exibidas para o usuário, para este fim, a função print jogo mais gol é chamada.

A função recebe como parâmetro os dados referentes aos resultados e confrontos, no interior da função uma cadeia de for's encontra o confronto em que ocorreu o maior número de gols. Uma variável 'maior', igualada a zero se torna o maior número de gols assim que este valor é encontrado. Para visualizar a informação de for's vai em busca de todos os resultados que obtiveram este 'maior' como número de gols. O processo é demonstrado na figura 15 abaixo.

```
for itens in data:
    for grupo in itens:
        for info in itens[grupo]:
            for confronto in info:
                for resultados in info[confronto]:
                    if resultados['RESULTADO TIME 01'] == maior or resultados['RESULTADO TIME 02'] == maior:
                        x = confronto.split()
                        for elementos in dados:
                            for chave in elementos:
                                for confrontos in elementos[chave]:
                                    for chapa in confrontos:
                                        if x[2] in chapa and x[6] in chapa:
```

Figura 15. Processo para encontrar confronto em que uma seleção marcou mais gols

Ao encontrar os resultados que possuem este valor 'maior' como número de gols, a função realiza o split na chave para separar as seleções por índice e em seguida busca estas duas seleções no arquivo dos confrontos, para que o usuário visualize além do resultado, o dia, horário e local em que este marco ocorreu. Após a condicional de verificação o print é feito, inclusive em caso de empate entre os confrontos neste quesito.

A última opção do menu principal permite ao usuário visualizar a formação das oitavas de final, esta opção assim como a anterior se torna disponível apenas com o cadastro de todos os resultados, em caso de cadastro pendente o usuário será avisado.

Os dados referentes a todos os arquivos são carregados e a função oitavas é chamada para realizar a formação desta fase da copa, ela não possui parâmetros. No interior da função os dados referentes ao arquivo das tabelas são carregados e as listas ordenadas na função de print das tabelas são usadas para geradas.

O processo de formação das oitavas é feito pela função formar oitavas, que recebe como parâmetro as listas ordenadas geradas na função de print das tabelas, recebendo duas de cada vez, uma de cada grupo, como mostra a figura 16 abaixo.

```
confronto1 = "{} X {}".format(lista1[0],lista2[1])
confronto2 = "{} X {}".format(lista1[1],lista2[0])
print(confronto1)
print("============================")
print(confronto2)
```

Figura 16. Processo para formação dos confrontos

Seguindo as regras da FIFA, o primeiro de um grupo joga com o segundo do grupo seguinte, por consequência o segundo de um grupo joga com o primeiro do outro, para realizar este processo os confrontos são montados de forma manual levando em consideração os índices da lista.

O programa aceita somente a formação de confrontos pelo número de pontos, por este motivo não existem condicionais de desempate para formação dos confrontos.

Ao sair de cada submenu, o programa retorna para o menu principal que rotaciona toda a aplicação, caso o usuário escolha a opção sete, um break é acionado e a aplicação é encerrada.

2.3 **Ordem de codificação**

A linguagem de programação Python exige que todas as classes e funções sejam criadas fora do programa principal, portanto todas as funções são criadas inicialmente, as duas classes utilizadas pelo programa, são definidas além disso a biblioteca JSON é importada para utilização na manipulação dos arquivos.

O programa principal inicia com o menu principal, sete opções de escolha são ofertadas, cada opção possui submenus que levam o usuário para cada função do código.

Listas de verificação e dados dos arquivos são carregados ao início de todo menu, para controle nos inputs. Condicionais são dispostas no início de cada menu para avisar ao usuário sobre a falta ou o completo preenchimento de algum cadastro.

A ordem dos menus é a seguinte: menu dos grupos(para cadastro, edição ou exclusão de um grupo), menu dos jogos (para cadastro, edição ou exclusão do dia, horário ou local de um confronto), menu dos resultados (para cadastro ou exclusão do resultado de um jogo), menu das tabelas(para visualização da ordem das seleções em um grupo por pontos, saldo de gols e gols marcados), menu de estatísticas (para visualização das médias de gols por grupo e geral, além do jogo em que uma seleção mais fez gols), menu das oitavas de final (para visualização dos confrontos formados e classificados, apenas com os número de pontos) e opção sete como saída.

Para criação do programa o Windows 11 Home baseado em 64 bits foi utilizado, o ambiente integrado de desenvolvimento utilizado foi o Visual Studio Code utilizando o Python 3.11.

3. Resultados e discussões

De forma geral diversos testes foram feitos para verificação da integridade dos arquivos,

correta edição do nome de uma seleção em todos os arquivos, somatório correto das médias e exibição do confronto em que uma equipe marcou mais gols.

Testes foram realizados para assegurar que o usuário não comprometesse a integridade das informações, diversas condicionais foram alocadas ao longo do programa para enviar avisos de erros ou de informações faltosas.

Foram averiguadas também a fluidez entre os menus e o tempo ou momento em que o arquivo era atualizado com uma nova informação enviada pelo usuário.

3.1 Dados de entrada

O usuário deve seguir primeiramente a opção que deseja no menu principal, após isso, ele será redirecionado para a opção escolhida, cada menu possui sua particularidade, condicionais são colocadas para evitar a entrada de dados repetidas, ou seja, que já estão presentes no arquivo.

Nos menus as entradas disponíveis são unicamente aquelas disponíveis como escolha, não existe limitações ou verificações para o dia, horário e local de uma partida. No menu principal caso a opção escolhida seja a de cadastro ele escolherá se quer cadastrar, editar ou excluir um grupo, é permitido no caso do cadastro a paralisação e continuação depois que o programa se encerre, se o usuário assim desejar.

Ao cadastrar uma seleção, caso esta tenha mais de uma palavra em seu nome, ele deve digitar o nome sem espaços entre as palavras devido ao método de busca das seleções por índice usando o split nas chaves. Exemplo, Estados Unidos deve ser digitado EstadosUnidos.

Para editar um grupo devem ser escolhidos o grupo e a seleção que será retirada e inserida neste grupo, para exclusão, basta que o usuário escolha o grupo que será excluído. Após qualquer atividade feita no menu dos grupos o programa deve ser reiniciado devido a problemas de fluidez.

Os dias, horários e locais de todos os confrontos de um grupo são cadastrados de uma única vez, o usuário escolhe qual grupo passará pelo cadastro não podendo cadastrar informações em um grupo que já possui seus jogos com as informações preenchidas.

Ele poderá editar o dia, horário e local de um confronto, escolhendo o grupo, o confronto e a informação que será alterada, a edição só é permitida se o cadastro existir, a exclusão é feita a partir de uma escolha, ele poderá apagar as informações de um confronto ou de todos, não podendo excluir informações que não foram cadastradas.

Os resultados estão apenas disponíveis com o cadastro de todas as informações relacionadas aos confrontos, ele escolhera o grupo e o confronto, e irá informar o número de gols marcados por cada equipe, não é permitido que se cadastre o resultado de um confronto que já foi previamente preenchido com seu resultado, nem que se acesse o cadastro dos resultados de um grupo que já possui todos os seus confrontos devidamente completos com relação ao resultado.

Para a visualização das tabelas é necessário informar apenas o grupo que terá sua tabela exibida, estatísticas e oitavas estão disponíveis como saída somente se todos os resultados estiverem cadastrados.

3.2 Dados de saída

Nos menus dos jogos e resultados é possível visualizar as informações referentes a todos os confrontos de um grupo informado pelo usuário, como mostram as figuras 17 e 18 abaixo.

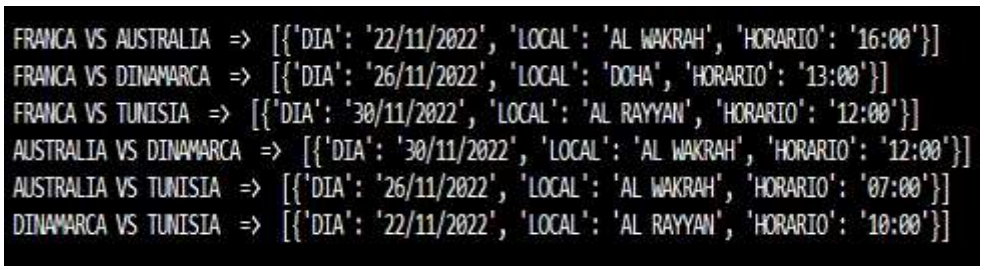

Figura 17. Exibição das informações dos confrontos do grupo D

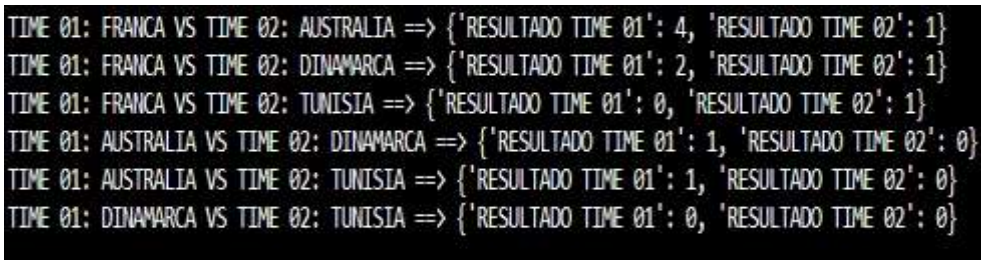

Figura 18. Exibição dos resultados dos confrontos do grupo D

Ao editar a informação de um confronto, a saída mostrando que a alteração foi feita é disponibilizada ao usuário para atestar a mudança feita.

O menu das tabelas disponibiliza ao usuário a visualização das seleções de um grupo em ordem pelo número de pontos, gols marcados e saldo de gols, como mostra a figura 19 abaixo.

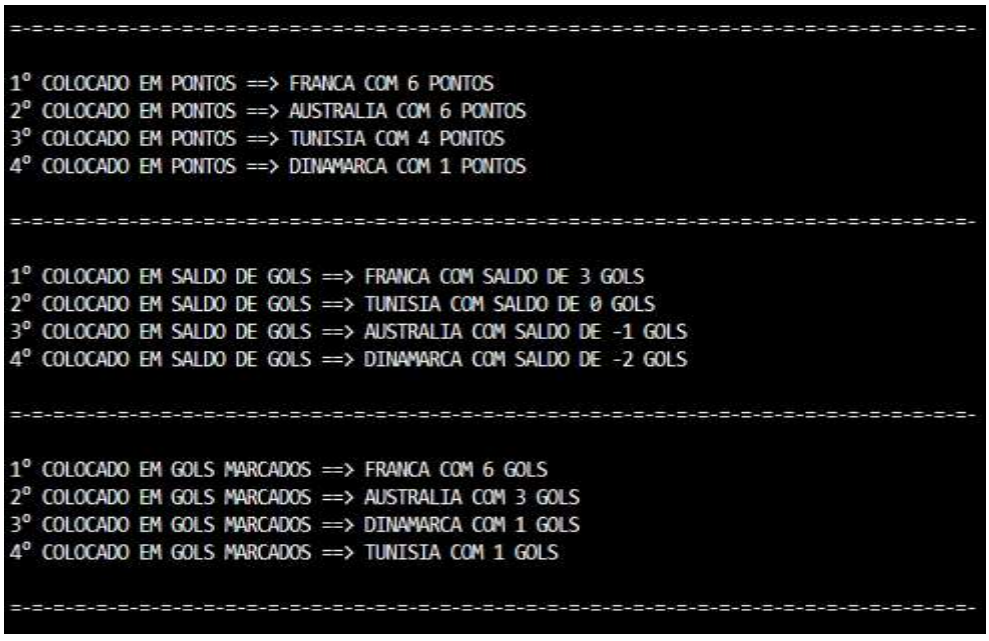

Figura 19. Exibição da tabela do grupo D

Um relatório de estatísticas é gerado caso o usuário assim queira, este, está somente

disponível com o cadastro de todos os resultados, a figura 20 abaixo exibe o processo utilizando como valores os números reais de gols da Copa do Mundo FIFA 2022

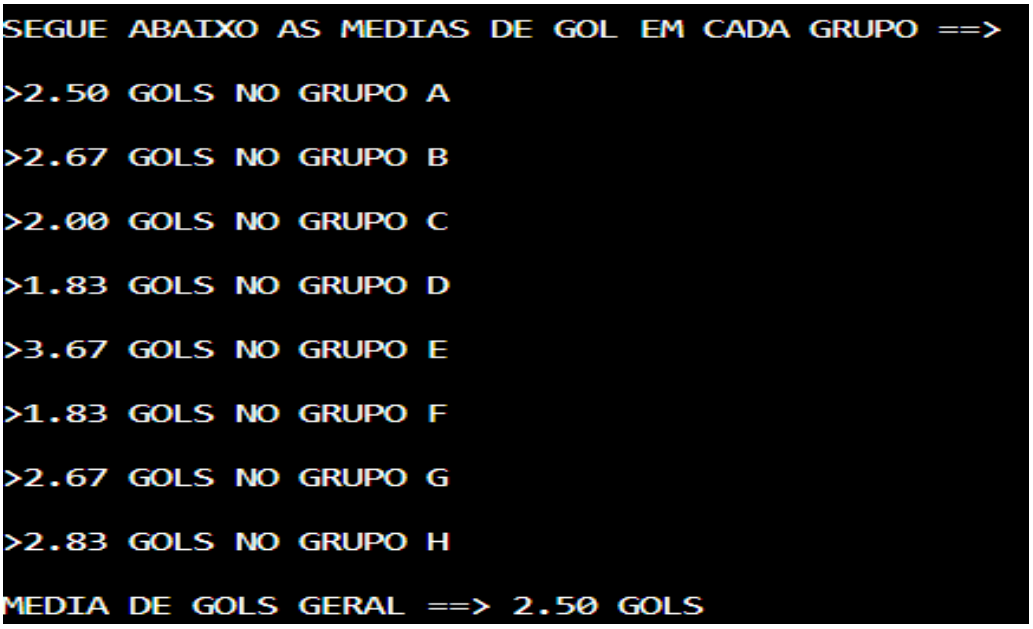

Figura 20. Médias de gol por grupo e geral

Este menu de estatísticas também permite que o usuário visualize a partida em que uma seleção marcou mais gols, além de todas as informações disponíveis sobre ela, a figura 21 exemplifica a situação, adicionando um elemento hipotético de empate em um quesito como este, atestando que mesmo que esta situação ocorra a informação será exibida sem incoerências.

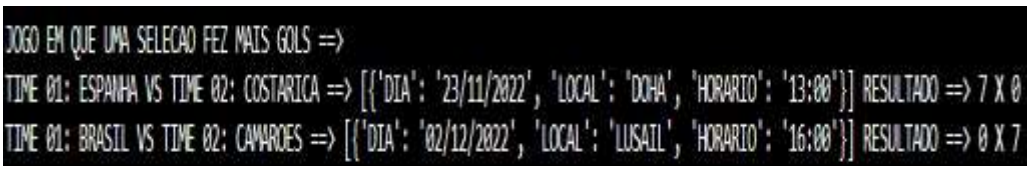

Figura 21. Jogo em que uma seleção fez mais gols

O menu das oitavas exibe os confrontos prontos com os classificados apenas pelo número de pontos, por este motivo incoerências podem ser encontradas na saída desta informação.

3.3 Testes e erros

Diversos testes foram feitos para verificar a fluidez dos menus, integridade dos dados cadastrados, correto processamento das informações de saída, e do somatório para o saldo de gols, pontos e gols marcados.

Nos menus de edição dos grupos foi observado se ao mudar o nome de uma equipe a mudança se fazia presente em todos os arquivos, nenhum erro foi encontrado neste quesito, além disso observou-se a relação entre os menus e os arquivos, para atestar a correta informação enviada ao usuário e recebida por ele.

Nenhum erro foi encontrado nas opções de cadastro, edição e exclusão, além disso as informações de estatísticas e da tabela foram verificadas para averiguar o correto processamento e somatório dos dados.

Problemas de fluidez foram encontrados no menu principal, situações de travamento que necessitavam o reinicio do programa foram encontradas, entretanto não foi observada perda da integridade dos dados devido a este problema.

Por fim, o programa não apresenta um sistema de desempate que vá além do somatório dos pontos para entregar um correto chaveamento das oitavas para o usuário, todavia não foram encontrados erros nas ordenações realizadas para formação das oitavas utilizando somente os pontos, levando em desconsideração os empates ocorridos neste quesito. Por fim não foram encontrados erros na formação das tabelas com as informações de cada seleção de um grupo.

4. Conclusão

O aplicativo fornece ao usuário todas as opções de cadastro possíveis visando a primeira fase da Copa do Mundo, erros de fluidez foram encontrados e o correto processamento de desempate para formação das oitavas não é realizado.

Pensando em aprimoramentos e melhorias em futuras versões, a aplicação poderá suportar um sistema de desempate completo que leva em consideração todos os quesitos utilizados pela FIFA, além de possibilitar ao usuário não somente o cadastro e as estatísticas da primeira fase do torneio, mas de todas as fases até o final.

No momento de cadastro das seleções a liberação total do cadastro para seleções com mais de uma palavra em seu nome deve ser introduzida para facilitar a usabilidade do programa pelo usuário.

O processamento deve ser melhor modularizado para gerar fluidez no menu principal, evitando incômodos aos usuários e melhores sistemas de visualização devem ser fornecidos para aumentar a disponibilidade de informações exibidas ao usuário.

5. Referências

Disponível em: https://www.lance.com.br/copa-do-mundo/como-surgiu-a-copa-do-mundo.html#:~:text=Evento%20come%C3%A7ou%20em%201930%20e%20est%C3%A1%20na%2022%C2%AA%20edi%C3%A7%C3%A3o%20do%20Qatar&text=LANCE!,08%2F12%2F2022&text=Criada%20em%201930%2C%20a%20Copa,para%20o%20torcedor%20grandes%20lembran%C3%A7as. Acesso em:11/dez/2022