RELATÓRIO E-FOLIO B

Nome: Rui Miguel Andrade Martins
UC: Programação por objetos
Número: 2002807

Relatório:

Comecei por criar a classe base definida no enunciado, a InfoJogo, tendo em conta que teria de colocar nestas classes os atributos e as funcoes que iriam ser utilizadas nas outras duas classes, a InfoJogoBasquetebol e InfoJogoFutebol. Portanto, coloquei os atributos sport(para definir o desporto em consideração), nome da equipas, score das equipas, uma estrutura de dados para identificar os jogadores, 2 listas para os titulares de duas equipas, 2 listas para os suplentes de ambas as equipas, e 2 iterators para as listas. Considerei estes os atributos comuns aos 2 desportos, e foram os suficientes para cumprir os objetivos pedidos no enunciado. De seguida, ainda na InfoBase, desenvolvi as funcoes que iriam ser partilhadas por ambos os desportos, tal como os getters e os setters dos atributos, além das funções para dar display ao jogadores e às informações do jogo. No construtor desta classe apenas peço os nomes de ambas as equipas, e nos construtores das classes futebol e basquetebol apenas corro um setSport para definir o desporto, pois os outros atributos são definidos a correr outras funcoes. A função mais complexa na classe base é a loadPlayers, onde vou alimentando a lista de titulares e suplentes, dependendo do desporto. Não gostei do facto de ter muito código repetido nesta função (gosto de seguir o principle DRY à risca), mas não consegui chegar a uma maneira de isso não acontecer. Comecei por criar esta funçao em ambas as classes basquetebol e futebol, o que gerava ainda mais repetição de código, por isso decidi meter esse excerto de codigo na função de classe, e assim instâncias de ambas as classes conseguem correr essa função à mesma. Decidi usar ficheiros para inserir as equipas, pois acho que seria bastante trabalhoso para o user estar a inserir um mínimo de 22 jogadores linha a linha no programa, e acho que não era esse o objetivo do programa. As funções transform que coloco, são apenas para colocar o input do user tudo em minuscula, para não haver erros na abertura do ficheiro, pois os ficheiros txt são normalmente todos nomeados em lowercase. Nunca tinha lidado com ficheiros em C++, por isso também aproveitei para aprender. Comecei por usar FILE *f como em C, mas depois de alguma pesquisa vi que o IFSTREAM era bastante mais fácil de usar, e optei por aí, também com o objetivo de desenvolver novas skills. Para dividir o input usei o strtok, e apesar de ser uma função de C, funcionou como eu pretendia aqui. Preferia ter usado os metodos find() e substr() para dividir a string de input, visto estar em trabalhar em C++, mas passado muito tempo de trial and error não consegui meter esse excerto a correr com essas funcoes, daí a razão de ter decidido usar funções de C como o strtok e o atoi, mas o programa corre como esperado à mesma. Uso o strtok para dividir as funções, crio um struct do tipo player para criar um novo player, e preencho a struct usando os tokens do strtok, correndo esse código o número de vezes que o user pede. Na classe InfoJogoFutebol, coloco os atributos sub, sentOff e playerScore, visto que são exclusivos ao desporto futebol, e também crio listas de subs, sentOffs, e playerScores, para poder guardar e demonstrar no final do programa, como é pedido no enunciado. Nesta classe, criei a uma função para

inserir substituições ou expulsões, dependendo do input do user. Novamente, usei strtok para dividir o input em tokens, e criar uma struct do tipo sub ou sentOff, dependendo se no 2º token foi encontrado o # ou não. Tambem peço ao user se deseja inserir a sub/expulsão na equipa A ou B, e dou push_back da struct para a respetiva lista. Tambem criei uma função para inserir os golos no jogo, indo buscar o total de golos no jogo usando os getScore, e correndo um loop essa quantidade de vezes, criando um struct do tipo playerScore, e usando o strtok para dividir o input e preencher a respetiva struct, dando no fim push_back para a lista. Tambem criei uma função setTeamsScore para usar os setters para definir os scores, para não cometer os mesmos erros do eFolioA, que recolhi o input diretamente nos metodos de acesso, sendo essa uma má prática. Apesar de usar using namespace std ser uma má prática, fi-lo no eFolioA sem nenhum tipo de problema, por isso voltei a fazê-lo. Todas as outras funções nesta classe são meramente para dar display a certos atributos do jogo, para cumprir os requerimentos definidos no enunciado. Na função InfoJogoBasquetebol, coloquei os atribuitos struct timePlayed e pointsScored, para guardar os dados sobre o tempo de jogo jogado de cada jogador, e também os pontos marcados, criando também listas para essas structs e os respetivos iteradores. Na função setTimePlayed, peço ao user para inserir quantos jogadores participaram no jogo(sendo o min 10) e corro um loop esse número de vezes, criando em cada loop uma struct timePlayed, e preenchendo a struct com os tokens criados pelo strtok, a partir do input do utilizador, colocando no fim cada struct na lista. Faço uma condicional para error handling, caso os minutos ou segundos sejam acima de 60. Tambem crio uma função setPlayerScores para o user introduzir os pontos marcados por um determinado jogador, pergunto ao user quantos jogadores marcaram nesse jogo, e correndo um loop essa quantidade de vezes, criando uma struct do tipo pointsScored, usando um strtok para dividir o input, e preenchendo esse struct, guardando-o numa lista. Para error handling, uso os getScores para determinar a soma total de pontos no jogo, e caso a soma de pontos inserida pelo user seja diferente, limpo a lista criada ate aí, e corro a função novamente. Tambem crio a função setTeamsScore, igual à da classe InfoJogoFutebol, apenas com error handling para não correr os setters enquanto os resultados das duas equipas não sejam diferentes. Todas as outras funções desta classe são para dar display dos atributos do jogo. Apesar do programa fazer tudo que pede no enunciado, apenas o faz se os inputs forem todos corretos, caso contrário irá dar erros, ou não correr como é esperado. Devido a má gestão de tempo da minha parte, não consegui fazer todas as validações que pretendia, por exemplo, o objetivo seria apenas deixar introduzir número de jogadores nos golos/substituições, que tivessem na lista de titulares, ou que se tivessem entrando no decorrer do jogo, pois o programa como está desenvolvido, deixa inserir qualquer número, mesmo que esse número nem exista na equipa. Apesar de isso me parecer fácil de implementar, iria ser trabalhoso e acrescentar bastantes linhas ao programa, o que me iria impossibilitar de fazer o resto que pede no enunciado, por isso decidi fazer todos os requirements que o enunciado pede, e se o user inserir os dados reais do jogo, sem nenhum erro, o programa corre perfeitamente, e faz tudo o que é pretendido. Para finalizar, envio no ficheiro zipado 2 ficheiros txt de equipas de futebol(porto e benfica) e 2 ficheiros de equipas de basquetbol(lakers e bulls) para o professor fazes os testes que pretender.