Arquivos Binários - Estatísticas NBA

1 Descrição

Neste VPL você deverá trabalhar com arquivos binários e estruturas (structs). Você deverá implementar algumas funções para computar estatísticas e obter informações compiladas com base em dados da temporada 2018-2019 da NBA (as informações foram tiradas daqui). Neste VPL você não precisará enviar um *programa* (você não deve enviar uma função *main*), só as implementações das funções solicitadas. Note que há um arquivo `nba_stats.h` no qual são apresentados os cabeçarios das funções e descrições do que elas devem fazer. As funções devem ser implementadas no arquivo `nba_stats.c`. Não altere o arquivo `nba_stats.h`.

Abaixo é fornecido um "passo a passo" de como proceder para este VPL.

1. Baixe o arquivo zip disponível e descompacte o mesmo em uma pasta do seu computador.
2. Dê uma olhada no arquivo `jogadores.ods` e no arquivo `nba_stats.h` para ententer o que se pede.
3. Compile e execute o programa resultante do arquivo `gera_binario.c`. Este arquivo contém o código responsável por ler o arquivo `jogadores.csv` e criar o arquivo `jogadores.dat`, que seu programa deve ler para trabalhar neste VPL (lembre-se, o VPL é de arquivos binários). O arquivo `csv` e o executável gerado devem estar na mesma pasta para funcionar. Não altere os arquivos `gera_binario.c` e `jogadores.csv`, tampouco o arquivo obtido `jogadores.dat`.
4. Além dos arquivos que eu mencionei antes, ele contém também: os arquivos `nba_stats.c` e `nba_stats.h` e um arquivo `main.c`. Você deve implementar as funções no arquivo `nba_stats.c` e pode (eu diria que *deve*) usar o arquivo `main.c` para ir testanto suas funções. Note que o arquivo `main.c` está quase vazio, porque você tem que fazer seus testes nele. Para testar as funções, use o arquivo `jogadores.csv` como referência de consulta e faça chamadas às suas funções.
5. O formato do arquivo `.dat` que seu programa deve manipular (basicamente ler) é o seguinte: ele tem um único inteiro no seu início que indica quantos jogadores (registros / estruturas) estão armazenados no arquivo. Após esse inteiro estão armazenados os registros dos *n* jogadores indicados pelo inteiro inicial. Este valor inteiro no começo do arquivo é útil pois permite saber: quantos registros devem ser lidos e, consequentemente, que tamanho de vetor deve ser alocado para armazená-los.

Se você tem dúvidas, abaixo estão algumas respostas para perguntas frequentes.

- **Por que você não preciso enviar uma *main*?** Porque eu, professor, fiz uma *main* que irá chamar as funções que você implementou, para testar se elas estão certas ou erradas. Você não terá acesso à minha *main*, ela fica "escondida" :-)
- **Isso significa que eu não preciso de uma *main*?** Não! O ideal é que você baixe os arquivos .c e .h, assim como os arquivos de entrada, crie um projeto no seu computador (no Codeblocks, por exemplo) e crie uma *main* para ir testando suas funções ao longo do desenvolvimento. Quando achar que está tudo certo, envie as funções implementadas, sem a sua *main*.
- **Eu não consegui/tive tempo de fazer todas funções. Posso fazer uma submissão parcial?** Sim! Pode! Para que o programa possa ser avaliado e compilado, todas funções devem estar implementadas no arquivo .c. Mas pode ser "qualquer" implementação. Então, se você não conseguiu (ou não teve tempo) de fazer todas funções, "implemente" as funções que faltam simplesmente colocando um "`return tipo;`" na sua implementação (onde tipo é o tipo de retorno pedido na função).
- **Eu ainda tenho dúvidas...** Procure os monitores e o professor, nós estamos aqui para lhe ajudar. Não deixe o conteúdo acumular e lembre-se: não existe "dúvida boba". Se você tem uma dúvida, ela é importante e tem que ser solucionada!