Tarefa 9 – Árvore Binária de Busca (ABB) - PESO 1.7

Atenção: inclua cabeçalho com seu nome completo, RA e Turma, comentários e INDENTAÇÃO no programa. Coloque o nome do arquivo com o seu nome.

Faça um programa que leia uma sequência de dados de alunos como: RA, nome e 3 notas. Monte uma ABB com estes alunos em que a ordem na árvore deve ser pelo número do RA (um inteiro). Ao montar a árvore, lembre-se que não pode haver chaves repetidas em uma ABB, portanto, verifique se o usuário não tenta incluir estudantes com uma chave (mesmo RA) que já existe na árvore e não deixe que isso ocorra (mostre mensagem de erro e peça para entrar novamente com a chave). Faça essa verificação antes de pedir os demais dados dos aluno!

O programa deverá conter as seguintes opções (não mude a ordem das opções para facilitar os testes):

1) Inserir um aluno na árvore;
2) Imprimir a árvore atual;
3) Mostrar os dados dos alunos que foram reprovados;
4) Excluir da árvore todos os alunos que foram reprovados;
5) Mostrar todos os alunos com chave menor ou igual a um dado RA;
6) Iniciar uma nova árvore;
7) Sair.

Na opção 2 você deve imprimir a árvore na forma como foi feito no exercício 7, ou seja, no formato de árvore horizontal e mostrando somente a chave (o RA).

Para a opção 3 você deve calcular a média das notas (o que pode ser feito logo após o usuário entrar com as 3 notas), guardá-la na árvore (como mais um dado do usuário) e usá-la nas opções necessárias (3 e 4). Considerar que um aluno é aprovado se sua média for maior ou igual a 6.0.

Nesta opção 3 você deve mostrar todos os dados de cada aluno reprovado, incluindo a sua média final.

Na opção 5 deverá ser solicitado ao usuário que digite um RA para que seja buscado na árvore todos aqueles que apresentem RA menor ou igual àquele digitado. Esta busca é interessante para verificar quais alunos são mais antigos, por exemplo (aqueles que apresentam menor RA). Preste atenção porque, como a árvore é ordenada, você não deverá fazer busca em toda a árvore nesta opção. **Otimização é primordial aqui, já que é uma ABB!!**!

As opções 6 e 7 implicam, necessariamente, em deslocar todos os nós alocados! O programa só deve ser finalizado quando o usuário digitar a opção 7. A opção 6 destrói toda a árvore criada (se existir alguma) e permite que seja iniciada uma nova ABB. Isto permite que sejam realizados vários testes no programa sem a necessidade de sair dele.

Especificações:

- O programa deverá ser implementado utilizando árvores binárias de busca. A não utilização dessa estrutura implicará em nota ZERO.
- Toda vez que alocar um espaço de memória, não esqueça de desalocá-lo antes de finalizar o programa.
- O programa deverá ser implementado em C.
- Não é permitido o uso de saltos como “break”, “goto”, “exit”, etc..
- Todas as variáveis deverão **obrigatoriamente** serem definidas no início de cada função que as utilizarão (ou como variáveis globais, dependendo do caso).
- Não será aceita entrega de exercício de qualquer outra forma que não seja pelo Classroom até a data limite de entrega.
- Erros de compilação implicarão em nota zero.
- Tentativas de fraude implicarão em nota zero para todos os envolvidos.
- Serão avaliadas corretude, eficiência e otimização do código.

- A não observância a qualquer item incluído nestas especificações implicarão em perda de pontos ou até em pontuação zero no exercício.
- Você deverá postar apenas o código do programa (.c) compactado (.zip) no link disponível no Classroom. Exercícios enviados por qualquer outro meio ou postados após a data limite não serão corrigidos. Coloque o **SEU NOME** no nome do arquivo. **Não poste o executáve**l, senão seu exercício nem será corrigido (mesmo que esteja com o código C junto, pois o Google entende que o arquivo compactado contém vírus por conter um executável).
- **Verifique se o código que foi postado é o código correto, pois não será permitida a entrega após a data e horário limite estabelecido.**