Exercício – Árvore AVL de Jogos

Este exercício irá utilizar a estrutura de jogos do exercício de catálogo de jogos. Você pode reutilizar os arquivos relativos aos items de jogos do exercício anterior.

Porém, será necessário ler os dados dos jogos de um arquivo exerno, chamado “CSV.csv”, que estará presente no próprio sistema. Portanto, você deverá ler este arquivo para criar sua árvore, independente da entrada que o exercício irá fornecer pelo *stdin*.

Neste exercício o programador deve criar um catálogo de jogos como **ARVORE AVL.**

O exercício pode ser feito em duplas. Porém deve haver um documento com o nome e número USP dos integrantes da dupla. Apenas um integrante da dupla deve submeter o exercício.

Nota: O “CSV.csv” está em UTF-8, logo há 3 bytes no início do arquivo “informando” isso. Para ler os dados o aluno não deve considerar esses 3 primeiros bytes. Uma das formas de desconsiderar esses 3 bytes é usando “fseek”.

Objetivo

O objetivo deste exercício prático é estimular os estudantes a se familiarizarem com o comportamento e a lógica associados à estrutura de dados **ARVORE AVL**.

Queremos que os alunos se habituem com a implementação dessa estrutura e consigam entender as funções básicas, que garantem a execução adequada desse TAD, em diferentes contextos.

Para acostumar os alunos com conceitos de modularização e boas práticas de escrita de código, será exigido o uso de múltiplos arquivos .c e .h no projeto, bem como a construção de um arquivo Makefile, responsável por gerenciar a execução do programa.

Descrição

O Programa deve rodar até receber um “F” de entrada. Quando receber o “F” o programa deve imprimir a arvore no formato que foi pedido E desalocar TODA a memória alocada dinamicamente.

O CSV.csv terá o seguinte formato: Nome do jogo (coluna 1), ano de lançamento (coluna 2), produtora (coluna 3); sendo cada linha um novo jogo. TODOS os jogos do CSV devem ser adicionados a arvore AVL, antes de qualquer possível remoção.

Cada nó da arvore é uma estrutura de dados “jogo” que foi definida no exercício de catálogo de jogos. Ou seja, uma estrutura com Nome, Ano, Produtora. A arvore sempre deverá ser balanceada pela data de lançamento dos jogos. Todos os nós a esquerda do nó pai terão uma data de lançamento menor que a do nó pai, enquanto todos os nó a direita uma data de lançamento maior. IMPORTANTE: caso as datas de lançamento sejam iguais o programa deve ordenar por ordem alfabética (pelo nome do jogo) crescente.

Ou seja o nó com ‘‘nome do jogo’’ menor ira para a esquerda, e o maior para a direita.

Resumindo: O programa deve sempre comparar a data de lançamento do jogo caso elas sejam iguais deve comparar o nome do jogo. Sempre colocando o menor para a esquerda e o maior para a direita.

A primeira entrada (um inteiro) define a forma como vamos imprimir a arvore AVL no final da execução:

1 – Pré-ordem

2 – Em ordem

3 – Pós-ordem

Nota: para simplificar a impressão, o programa deve imprimir apenas o Nome completo do jogo.

As próximas entradas serão referentes a remoção de algum nó da arvore após TODOS os jogos alocados e balanceados.

A remoção deve seguir a regra de substituir o nó removido, quando necessário, pelo **maior nó da sub-árvore da esquerda**.

O Usuário pode requerer a remoção de algum jogo, para isso ele irá inserir um número referente ao ano de lançamento do jogo. TODOS os jogos com essa data de lançamento devem ser excluídos. O programa deve excluir um jogo por vez e fazer o rebalanceamento Da arvore. Caso não haja nenhum jogo com essa ano o programa não fara nada.

Lembrete, o usuário pode pedir para remover quantos jogos ele quiser. O programa só deve ser finalizado quando receber o ‘‘F’’.

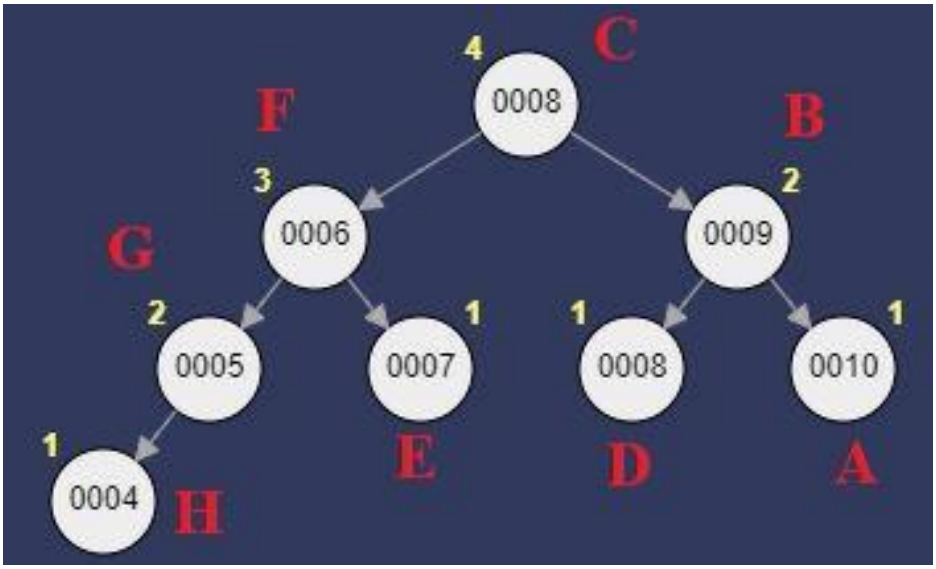
O programa só irá imprimir a arvore após o ‘‘F’’ (no formato especificado no início, pré-ordem [1] , em ordem [2] ou pós-ordem [3])

Exemplos:

Para todos os exemplos abaixo o CSV lido foi:

	A	B	C	D
1	A	10	L	
2	B	9	M	
3	C	8	N	
4	D	8	O	
5	E	7	P	
6	F	6	Q	
7	G	5	R	
8	H	4	S	
9				

Ou seja, a arvore INICIAL, sem nenhuma remoção é:



Exemplo 1 –

Entrada:

2

F

Saida:

H

G

F

E

C

D

B

Exemplo 2 -

Entrada:

2

8

F

Saida:

H

G

F

E

B

A

Observações da avaliação

A avaliação do seu programa será feita além do resultado da plataforma run.codes. **Portanto, ter um bom resultado com os casos de teste, não será suficiente para garantir a nota máxima** e **nem a aprovação do exercício.**

Caso seu projeto não satisfaça os pontos exigidos nos objetivos e explicitados nas observações de implementação, **sua nota poderá ser reduzida ou ser desconsiderada**.

Cópias de código entre alunos, acusadas pela plataforma, resultarão imediatamente em zero aos dois ou mais alunos envolvidos.