TRABALHO

PRÁTICO

Aluno: Amaury Mário Ribeiro Neto

Matrícula: 17 205 0071

Disciplina: Algoritmos e Estruturas De Dados II

Professor(a): Sofia Costa Paiva

Ciência da Computação/UFSJ

1º SEMESTRE – 2018

Introdução

A prática consiste basicamente na criação de uma lista telefônica de alunos que deverá armazenar o nome, a matrícula e o número de telefone de cada aluno de forma ordenada, utilizando para tal fim uma estrutura de dados do tipo Árvore AVL.

O programa deve realizar as seguintes operações: Criar lista, Inserir, Remover e Buscar alunos; além dessas o menu disponibilizará também a opções de Impressão da lista completa e a opção Sair. A chave usada para a ordenação será o número de matrícula de cada aluno, onde através dela pode-se realizar uma busca dos dados relacionados ao cadastro desejado.

A impressão da lista será realizada no próprio terminal simulando uma árvore AVL "deitada". Essa impressão é feita automaticamente a cada comando de inserção ou remoção adotado e poderá ser feita também pelo menu exibido no terminal.

(O programa não possui interface gráfica sendo focado na implementação e realizado inteiramente via terminal.)

Implementação

O programa é composto basicamente de 4 arquivos.

O arquivo *arvore.h*, no qual ocorre a declaração das funções e da struct contendo os apontadores utilizados, além da struct responsável pelos dados de cada aluno na árvore.

No arquivo *arvore.c* são implementas todas as minhas funções e os apontadores da árvore, ambos comentados de forma explicativa e organizada.

O arquivo *main.c* estão os códigos responsáveis criação do menu principal cujo objetivo principal é a utilização das funções implementas nos arquivos árvores para a realização das operações necessárias.

Além desses arquivos também está incluído o arquivo *Makefile* responsável por executar os códigos de compilação do programa. Para compilar o programa no terminal basta acessar a pasta onde estão localizados os arquivos em seguida digitar: "*make"* (sem as aspas) e pressionar a tecla enter. Caso seja necessário, a exclusão dos arquivos compilados (arquivos de extensão ".o") poderá ser feita pelo comando "*make clean"* (sem as aspas)

Funções

As principais funções utilizadas na implementação do programa são:

- **inicializar:** Função básica que realiza a inicialização da árvore cujo valor inicial será NULL;

- **inserir_arvore:** Recebendo por parâmetro os valores digitados pelo usuário, esta função irá buscar (por meio do número de matricula) o local adequado para a inserção do cadastro, cuja operação será feita por uma função auxiliar (criar_no será citada futuramente). Após encontra-lo ocorre o balanceamento (também feito por funções auxiliares) da árvore de alunos e a sua impressão.

- **remover_pilha**: Após o usuário inserir o número de matrícula do aluno que deseja remover, a função irá buscar a localização do nó(matrícula) dentro da árvore e removê-lo caso seja encontrado. Após a remoção é realizados os balanceamentos necessários além da impressão da lista completa.

- **buscar_arvore:** Responsável por realizar a busca, utilizando o número de matrícula inserido pelo usuário, dentro da árvore. Ao encontrar a função faz a impressão dos dados relacionados àquela matricula registrada na lista telefônica, caso contrário um mensagem de aluno não registrado será mostrada.

- **printar_arvore:** Esta função faz a impressão de todas as matriculas registradas usando a notação de parênteses para mostrar sua localização na lista em forma de árvore AVL. Sua ordem é feita por caminhamento central, ou seja, ocorre de forma recursiva.

Funções auxiliares:

- **fator_balanceamento:** Como o próprio nome sugere, essa função tem o objetivo de calcular o fator de balanceamento do nó recebido por parâmetro em seguida retorna-lo. O fator de balanceamento é feito pela subtração das alturas dos nós filhos.

- **altura:** Retorna a altura do nó recebido por parâmetro. Esse valor é apontado pelo ponteiro “alt”.

- **maior**: Função básica que recebe dois valores inteiros e retorna o maior dos dois.

- **criar_no**: Esta função é responsável por auxiliar na inserção de novos cadastros, salvando os dados do aluno na struct correta e a atualização dos apontadores (“esq” e “dir”).

- **menor_filho:** Seu objetivo é a busca do menor filho dentro da árvore, após encontra-lo, ele é retornado. Serve como auxiliar para a operação de remoção de nós com 2 filhos.

- **balanceamento_dir:** Responsável por verificar se o tipo de rotação para a direita utilizada para o balanceamento será simples ou dupla .

- **balanceamento_esq:** Bem parecida com a anterior, porém o tipo de rotação a escolher será esquerda (simples ou dupla).

Funções de Rotação:

- **rotação_sd:** Sua função é realizar o rotação do nó deseja para a direita.

- **rotação_se:** Sua função é realizar o rotação do nó deseja para a esquerda.

- **rotação_dd:** Rotação dupla, através das rotação simples esquerda e direita, respectivamente, ela realiza a rotação dupla direita

- **rotação_de:** Outra rotação dupla, porém para o lado esquerda, invertendo a ordem das funções simples a serem usadas. Rotação simples para direita em seguida a rotação simples para a esquerda.

Função Main:

A função Main é responsável basicamente por criar o menu principal para uso do usuário, além da utilização das funções necessárias, para a realização das operações desejadas pelo usuário

Complexidade

A operação de busca tem complexidade **O(log n)** onde n é o número de nós da árvore. Sua velocidade é justificada devido ao fato de seus nós estarem sempre balanceados, onde após a comparação e realizado uma recursão que diminuirá a cada execução o número de nós a serem analisados pela metade até a sua localização.

Conclusão

A prática foi um grande desafio, exigindo bastante conhecimento teórico do aluno sobre o assunto. Grande parte da dificuldade ocorreu devido ao curto prazo de conclusão, este que se agravou com a necessidade de realizar outros trabalhos e avaliações pertencentes as demais disciplinas do curso.

Durante a implementação das funções foi posto em prática todo o entendimento teórico visto em sala de aula. Algumas funções, como as responsáveis pelo balanceamento, se mostraram um grande obstáculo a ser superado, o que acabou contribuindo para a fixação do conteúdo, além de efetivar a ligação entre teoria e prática.

Devido a gama de conhecimentos e experiências adquiridas através de inúmeras tentativas e falhas, o trabalho rendeu como um todo um enorme aprendizado, este que será de grande utilidade em disciplinas futuras.

Bibliografia

Livros:

- **The C Progamming Language Brian - W. Kernighan and Dennis M. Ritchie - Second Edition - 1988**

- **Linguagem C : Descomplicada - André R. Backes - 2017**