Estruturas de Dados

Trabalho Prático de implementação – Listas encadeadas

1. Regras

- O trabalho deverá ser desenvolvido individualmente, em linguagem C.
- O trabalho vale 25 pontos.
- Trabalho copiado vale 0(zero).

O professor poderá realizar arguição individual sobre o trabalho a fim de averiguar a compreensão de cada aluno sobre o trabalho realizado podendo a nota do trabalho ser substituída pela nota da arguição.

2. Introdução:

Listas encadeadas ordenadas são estruturas de dados lineares em que as operações de busca exigem, nos piores caso, n comparações (onde n é o número de elementos da lista). Diversas outras estruturas de dados foram criadas para melhorar essa característica, como, por exemplo, árvores binárias, árvores B, tabelas hash, etc. Nesse curso ainda veremos árvores binárias. Há duas implementações clássicas de listas: simplesmente encadeadas e duplamente encadeadas. Nesse trabalho você deverá desenvolver um programa que se utilize de listas encadeadas ordenadas para implementar uma agenda de telefones. Você pode escolher entre implementar simplesmente ou duplamente encadeada.

3. Requisitos Funcionais:

Desenvolver um programa de agenda de contatos. Cada registro de sua agenda deve armazenar o nome e uma lista de telefones de um contato (um contato pode ter vários telefones armazenados). Para cada telefone deve-se armazenar o número e o tipo. O tipo do telefone pode ser: celular, casa, trabalho, principal ou outros. Assim, seu programa armazenará uma lista de contatos sendo que

Seu programa deve prover as seguintes funcionalidades:

- Incluir contato/telefone: o programa deve solicitar o nome do contato a ser incluído.
    - Caso já exista um contato com o nome informado, o sistema deve informar todos os telefones já existentes para aquele contato e perguntar se o usuário quer adicionar um telefone ao contato. Se a resposta for

positiva, deve solicitar o número e o tipo do telefone, acrescentá-lo à lista de telefones do contato e perguntar novamente se deseja acrescentar um telefone, até que a resposta seja negativa.

- Caso não exista um contato com o nome dado, deve ser criado um novo registro com o nome dado e solicitado que o usuário informe um telefone (número e tipo) a ser acrescentado na lista de telefones do contato criado. Em seguida deve perguntar se o usuário quer adicionar outro telefone ao contato e ficar em loop acrescentando novos telefones até que o usuário informe que não quer acrescentar novos telefones ao contato.

Dica: mantenha a lista de contatos já ordenada pelo nome ao inserir o contato. Isso facilitará a implementação do próximo requisito.

- Exibir Agenda Inteira: O programa exibirá todos os contatos da agenda (nome do contato e todos os telefones – número e tipo) ordenados alfabeticamente pelo nome.
- Buscar contato por nome: o programa solicita o nome do contato e exibe os dados de todos os contatos cujos nomes contenham a string informada. Por exemplo, se a string informada na busca for “Jose”, e existirem na agenda contatos com nome “Jose”, “Jose Luiz” e “Maria Jose” o sistema deve exibir os dados (nome e telefones) de todos eles (Dica: vejam a função strstr da biblioteca string.h). Não havendo contatos cujo nome contenha a substring dada, uma mensagem deve ser exibida informando que não foram encontrados contatos.
- Buscar contato por telefone: o usuário informa um número de telefone. Se o número informado for telefone de algum contato o sistema exibe o nome do contato. Caso contrário exibe uma mensagem informando que o número é desconhecido.
- Excluir contato por nome: O usuário informa um nome e o sistema exibe os nomes de todos os contatos cujos nomes contenham a string informada, oferecendo 3 opções para o usuário: (i) cancelar: nesse caso a função de exclusão é cancelada sem que nenhum contato seja excluído; (ii) escolher um contato a ser excluído: o usuário deve indicar um dos contatos para ser excluído e então o contato é removido; (iii) excluir todos os contatos selecionados. (Dica: caso cada contato tenha uma lista encadeada de telefones, lembre-se de desalocar os telefones antes de desalocar o contato.)

4. Requisitos não funcionais:

- Faça bibliotecas contendo as estruturas e métodos referentes a manipulação de listas.

Dica de estruturação: pense na possibilidade de criar uma estrutura “telefone”, uma estrutura/biblioteca de lista de telefones, uma estrutura “contato” e uma estrutura/biblioteca de lista de contatos.

5. Critérios de avaliação

Além da adequação aos requisitos acima listados, serão considerados na avaliação:

- Modularidade do código (divisão do código em funções e procedimentos);
- Qualidade dos comentários;
- Indentação do código.