Primeiro Trabalho Prático de Estrutura de Dados I

**** Rafael de O. Cozer e Aline Pontes Dorio**
DI/UFES**

Introdução

> O projeto visa simular uma versão simplificada de um sistema de cuidados com idosos, com a utilização de sensores, implementando o conceito de listas simplesmente encadeadas. O código foi elaborado utilizando-se diversas TAD's, atribuindo cada funcionalidade do sistema à uma biblioteca específica. A intenção final do projeto, era criar um sistema chamado “EDCare” a ser utilizado nos cuidados de idosos. Esse sistema deve ler sensores de idosos, onde consta sua temperatura, localização e se houve ou não uma queda. Com isso, cuidadores próximos aos idosos poderiam ser acionados automaticamente. Além disso, implementa uma rede de apoio, de forma que idosos amigos possam se ajudar.

> Nos tópicos a seguir, um pouco de cada uma dessas funcionalidades implementadas, seguindo a linha de raciocínio utilizada.

1. listaP e pacientes.h

Nessas TAD’s foram implementados os trechos de códigos relacionados diretamente ao tratamento dos pacientes. Faça-se saber aqui que o termo “idoso” foi substituído por “paciente”, pois pareceu mais plausível no momento da construção do código. Na listaP, foi trabalhada a lista de pacientes e na pacientes.h é onde começamos a rodar o código, através da inicializaEDCare.

2. listaC e cuidadores.h

Aqui, foram trabalhados os trechos relacionados aos cuidadores. Novamente, assim como anteriormente, a listaC foi onde criou-se as listas de cuidadores, atribuídas aos pacientes, uma vez que cada paciente tem sua própria lista de cuidadores disponíveis. Já na cuidadores.h, trabalhou-se sobre as estruturas dos cuidadores em si, armazenando suas structs.

3. listaA e amizades.h

Por fim, aqui trabalhou-se a questão das amizades entre os pacientes (ou também chamada de “rede de apoio”, à fim de averiguar os amigos disponíveis mais próximos de um idoso(paciente) em caso de febre baixa. Na listaA.h trabalhou-se com as listas de amizades, uma vez que cada idoso teria sua própria lista, assim como sua própria lista de cuidadores. Já na amizades.h, trabalhou-se em cima de sua struct e na criação das amizades em si.

Arquivos

É importante ressaltar aqui da importância da localização dos arquivos de entrada, que **devem** estar localizadas dentro da pasta “data” para que o código funcione de forma apropriada.

Como arquivos de entradas, utilizou-se os arquivos de teste disponibilizados pela professora, a saber: apoio.txt, cuidadores.txt e arquivos de cada idoso e cuidador (idoso1.txt, cuidador1.txt, etc). Como arquivos de saída, o programa deve gerar saídas para os idosos, com os sensores devidamente lidos (idoso1-saída.txt, idoso2-saída.txt, etc.).

Conclusão

O trabalho permitiu trabalhar bem o conceito de listas simplesmente encadeadas, uma vez que todo o projeto girou em torno dessa ideia. Além

disso, exigiu que se trabalhasse no quesito organização, uma vez que foram utilizadas uma grande quantidade de TAD's para este projeto.
Foi um trabalho desafiador e houveram muitas dificuldades.
Dessa forma, frente ao desafio, pôde-se aprimorar e fixar bem os conceitos vistos, de tal maneira à contribuir para evolução dos alunos enquanto cientistas da computação.
Como maiores dificuldades vistas, pôde-se apresentar principais:
Dificuldade relacionada aos #includes:
Em alguns momentos, principalmente no inicio do projeto, foram observados alguns problemas em relação ao uso dos #includes, para atribuir uma TAD à outra. Em certos momentos, isso estava causando loops de includes e, em outros momentos, o código simplesmente não aceitava o include da forma como estava sendo implementado.
Dificuldade com a liberação da memória:
Houve uma certa dificuldade na hora de liberar a memória alocada, uma vez que se optou por implementar os "free's" ao final, não durante a implementação do código. Sendo assim, teve-se que analisar com cuidado a forma como o código estava sendo executado, para que as liberações de memórias fossem encaixadas nos locais certos. Isso acabou consumindo um certo tempo e, talvez, a decisão de liberar a memória apenas no final não tenha sido tão interessante. Além disso, é possível que tenha ocorrido vazamento de memória frente à essa dificuldade.
Dificuldade ao calcular a distância entre os amigos, o que ocasionou em alguns erros ao gerar os arquivos de saída, nos trechos de "febre baixa", onde os amigos são acionados.

OBS.: É importante ressaltar que o código relatou bugs em arquivos de teste onde havia mais de uma linha vazia ao final do arquivo. **Para que funcione apropriadamente, deve haver no máximo uma linha vazia ao final do arquivo.** Não conseguimos tratar esta questão.
Além disso, ocorriam bugs em caso de haver um espaço vazio ao final das linhas onde liam-se os nomes de todos os cuidadores (em cuidadores.txt) e onde liam-se os nomes de todos os idosos (em apoio.txt). Esta questão conseguimos tratar e foi resolvida.
Por fim, não conseguimos implementar código em relação ao caso de "falecimento" de idosos, e em arquivos que contém esse trecho, ocasionam em seg fault no programa.

OBS.2.: Em alguns arquivos de teste disponibilizados, é apresentado algo semelhante ao seguinte:

febre baixa, acionou amigo i5
tudo ok
febre alta, acionou c9
tudo ok
febre baixa, acionou amigo i3
febre baixa pela quarta vez, acionou c9

Em nosso código, como orientado no enunciado do trabalho, a "febre baixa pela quarta vez" só é apresentada caso não haja alguma incidência de febre

alta entre esses episódios de febre baixa. No trecho acima, a febre baixa foi apresentada mesmo com incidência de febre alta no “meio”.

Bibliografia

https://pt.stackoverflow.com/
https://devdoc.net/
https://cboard.cprogramming.com/
https://www.geeksforgeeks.org/