EDCare

Nome: Matheus Santos de Moura
Matrícula: 2020102131

Relatório sobre o desenvolvimento do software EDCare

O trabalho proposto foi a criação de um programa que lê arquivos .txt, coleta dados e cria saídas .txt com o resultado da análise destes dados. Utilizando TAD's para organizar tipos, funções e structs, foi desenvolvido o EDCare, que analisa dados de: temperatura, latitude, longitude, se houve queda ou se houve falecimento de um idoso, e toma as medidas previstas nas instruções de implementação, ou seja, acionar um amigo em caso de febre baixa (menor ou igual a 37 graus), acionar um cuidador em caso de febre alta (maior ou igual a 38 graus) ou em caso de queda, bem como, caso não haja nenhum evento adverso, informar que está "tudo ok", também foi implementado a retirada de pacientes que falecem do sistema, bem como sua conexão com os demais pacientes.

De início, foi criado o TAD paciente, usando o conhecimento das aulas de Estrutura de Dados 1, com a definição da *struct Paciente*, com variáveis: *char * nome; float temperatura, double latitude, double longitude, int queda, int vivo, Lista * amigos, Lista * cuidadores, FILE * arquivo, FILE * arquivoSaida, int cont_febre_baixa* e as funções de *InicializaPaciente(*Aloca memória para um tipo paciente*)*, *ImprimePaciente(*Imprime o nome de um paciente*)*, *RetornaNomePaciente(*Retorna um ponteiro para o nome do paciente*)*, *Febre(*Retorna se o paciente teve febre alta (2), baixa (1) ou não teve febre (0*))*, *RetornaLatitudePaciente(*Retorna o valor da latitude do paciente*)*, *RetornaLongitudePaciente(*Retorna o valor da longitude do paciente*)*, *Caiu(*Retorno booleano para se o paciente caiu ou não*)*, *RetornaVivo(*Retorno booleano para se o paciente está vivo ou não*)* e *DestroiPaciente(*Libera a memória alocada para aquele paciente*)*, além das *RetornaAmigos(*Retorna a lista genérica de amigos daquele paciente passado como parâmetro*)*, *RetornaCuidadores(*Retorna a lista genérica de cuidadores do paciente passado como parâmetro*)*, *RetornaArquivo(*Retorna o ponteiro para o arquivo contendo as informações do paciente*)* e *RetornaArquivoSaida(*Retorna o ponteiro para o arquivo que deve ser impresso as informações do paciente*)*, após testar o funcionamento das funções, parti para a criação do TAD lista (lista encadeada com sentinela), que, inicialmente, não se tratava de uma lista genérica, coisa que foi corrigida depois, com as *struct Lista*, com variáveis: Celula * prim, Celula * ult, *e struct Celula*, com variavéis: void * item, Celula * prox e as funções *InicLista(*Aloca espaço para a lista genérica*)*, *InsereListaPaciente(*Insere um paciente em uma lista genérica*)*, *RetiraLista(*Retira um paciente de uma lista*)*, *DestroiListaPacientes(*Libera a memória alocada para a lista de pacientes*)* , *BuscaPaciente(*Devolve um ponteiro para um paciente contido em uma lista

) e *BuscaCuidador(*Devolve o ponteiro para um cuidador contido em uma lista*)* a princípio, após o teste das funções, a nova tarefa passou a ser criar o TAD Cuidador, com a *struct Cuidador*, com variáveis: char * nome, double latitude, double longitude, FILE * arquivo, e as funções *InicializaCuidador(*Aloca espaço para um ponteiro do tipo cuidador*)*, *ImprimeCuidador(*Imprime o cuidador*)*, *DestroiCuidador(*Libera a memória alocado para o tipo cuidador*)*, *RetornaNomeCuidador(*Retorna um ponteiro para o nome do cuidador*)*, *RetornaLatitudeCuidador(*Retorna o valor da latitude do cuidador*)* e *RetornaLongitudeCuidador(*Retorna o valor da longitude do cuidador*)*. Criado e testados os TAD's, foi criado a função *LePacientes(*Le os pacientes existentes no apoio.txt e estabelece a rede de amizades*)* e *LeCuidadores(*Le os cuidadores existentes em cuidadores.txt e estabeleca a rede de cuidado*)* dentro do TAD lista, bem como, as funções *LeLinhaPaciente(*Le uma linha do arquivo de informações ("nome do paciente".txt) e atualiza os dados do paciente*)* no TAD paciente e *LeLinhaCuidador(*Le a linha pela entrada do arquivo "nome do cuidador".txt e armazena essas informações no tipo cuidador*)* no TAD cuidador, testada as funções em arquivos especiais, foi criado ponteiros de arquivos dentro da *struct Paciente* e na *struct Cuidador*, bem como aberto os arquivos, usando seus respectivos nomes, dentro da função *InicializaPaciente* e *InicializaCuidador*, feito isso, foi criado a função de *Leitura(*Executa a leitura e atualização de todos os arquivos dos pacientes e cuidadores*)*, que faz a leitura de todos as linhas dos arquivos dos paciente e cuidadores, bem como toma as devidas providências em caso de febre ou queda.Após toda essa parte de análise de arquivos estar concluída, e os dados estarem armazenados corretamente, foi criada a função *CoeficienteDistanciaPaciente(*Retorna a distância entre dois pacientes*)*, no TAD paciente, bem como *CoeficienteDistanciaCuidador(*Retorna a distância entre um paciente e um cuidador*)*, que inicialmente, média quem estava mais próximo por um cálculo simples, (lat1-lat2)+(lon1-lon2), porém foi notado uma falha nesse método, e, mais para frente, a função foi ajustada para calcular a distância "real" dos membros passados como argumento, essa função foi usada na *AmigoMaisProximo(*Retorna um ponteiro para o amigo mais próximo do paciente passado como parâmetro*)* e na *CuidadorMaisProximo(*Retorna o cuidador mais próximo do paciente passado como parâmetro*)*, para encontrar o respectivo membro mais próximo. Ainda foram criadas funções para alterar o valor das variáveis, *contFebre* (Que conta quantas vezes aquele idosos teve febre baixa), a *AumentaFebreBaixa(*Aumenta o número de casos de febre baixa daquele paciente em um*)* e *ZeraFebreBaixa(*Zera os casos de febre baixa dquele paciente*)*, além da *RetornaFebre4x(*Retorno booleano se o paciente teve febre 4 vezes seguidas*)*, que, apesar do seu nome, retorna quando o paciente teve febre baixa 3 vezes, sendo a quarta encontrada dentro da função que analise os dados do nome.txt do paciente. Feitos os devidos ajustes na função *Leitura*, o trabalho já estava funcional!

O próximo desafio foi na criação do arquivo *makefile*, que só consegui graças a ajuda da monitoria do dia 14/02, onde os monitores me passaram como deveria ser

baixado o make no Windows (através do gerenciamento de pacotes, *Chocolatey*) e me auxiliaram na construção do mesmo.
Após o makefile concluído, utilizei a plataforma Repl.It, pois não encontrei suporte para o *Vangrind* no Windows, realizando assim a busca por vazamentos de meória e consertando os mesmos, até não haver mais nenhum.
Feito isso o trabalho foi concluído!
Os principais desafios foram na implementação do TAD lista genérica, que, antes de eu saber da possibilidade de existir uma lista genérica, estava cogitando fazer 2 TAD's de lista diferentes, um para pacientes e outro para cuidadores, também encontrei desafio na criação do makefile, problema esse que precisei da ajuda dos monitores para conseguir passar com êxito, e por último, levei algumas horinhas consertando vazamentos de memória detectados pelo *Valgrind*, principalmente na lista de amigos e cuidadores de cada paciente, na célula que apontava para o paciente/cuidador e nos pacientes excluídos, que não tinham sua memória liberada antes de saírem da lista, identificado essas falhas, a correção foi relativamente fácil e tranquila.
Gostei bastante da realização do trabalho, me lembrou o projeto de TCC que fiz no técnico, então tive um carinho especial pela ideia.

Bibliografia

- StackOverflow
- Repl.It
- Aulas de ED
- Monitoria de ED