

Estrutura de dados algoritmos



2018/2019

Docentes:

Ana Caraban

Karolin Baras

Filipe Quintal Trabalho elaborado por:

Alexandre Romão nº2047518

João Franco nº2046718

Paulo Ornelas nº 2046918

Rúben Rodrigues nº 2046018

Índice

1. Introdução
2. Problema
3. Solução proposta
4. Utilização da aplicação
5. Funcionalidades
6. Vantagens / desvantagens da solução desenvolvida
7. Conclusão e trabalho futuro
8. Referências
9. Introdução

Este trabalho teve como objetivo criar um programa utilizando C++, este projeto deverá trabalhar da forma mais simples e eficiente possível de modo a facilitar ao máximo a análise de quem o usa.

Neste relatório pretendemos que o funcionamento do programa seja percebido na melhor maneira.

Neste trabalho o grupo pretende não só esclarecer o nosso ponto de vista como também exemplificar algumas funcionalidades do programa.

1. Problema

O projeto que nos foi proposto consiste na criação de um programa na qual este simule uma cantina. Este mesmo programa deve executar todas as operações básicas de funcionamento duma cantina, de modo a que qualquer utilizador consiga utilizar o programa sem qualquer problema.

As operações que foram dadas como obrigatórias foram:

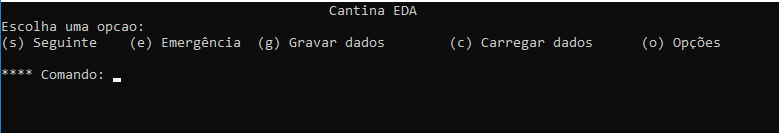
* Organização por grupos
* Fila de entrada da cantina
* Criação de refeições
* Faturação
* Mobilidade dos utilizadores da cantina

1. Solução proposta

Na primeira entrega do projeto o grupo decidiu implementar o que era pedido e implementar algumas das funcionalidades que seriam precisas mais tarde. Assim, a primeira entrega, usando vetores alocados dinamicamente, serviria de base para a implementação do projeto utilizando listas ligadas, o que mais tarde, verificou-se como uma decisão acertada. Nesta nova implementação, a a maior questão era a manipulação de dados sem perder a ligação entre eles, pois com a utilização de listas ligadas é de extrema importância nunca perder referência ao início de dita lista.

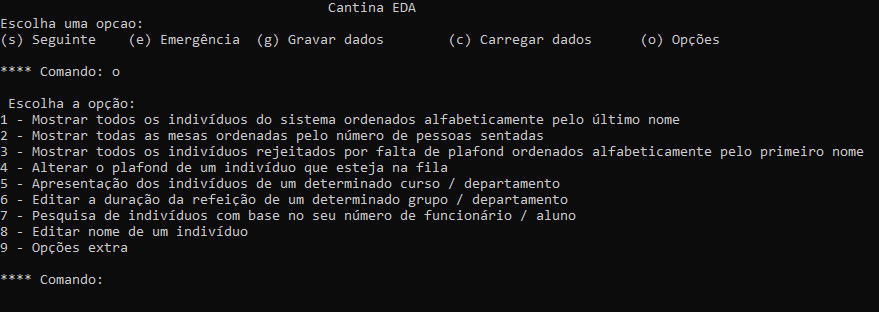
3. 1. Soluções adicionais

De modo a complementar o trabalho decidimos inserir/criar novas funcionalidades, para que este não se limitasse a uma simples cantina daí melhorarmos a estética como também a qualidade do código. Para tal inserimos as seguintes funcionalidades:

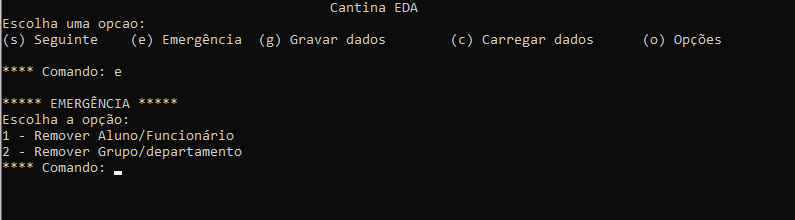
* Preencher a fila com novas pessoas;
* Retirar toda a gente da cantina;
* Alterar o curso de um grupo de alunos;
* Mostrar todas as refeições que a cantina teve;
* Mostrar a pessoa com maior plafond, menor plafond e o plafond médio das pessoas na cantina e fila;
* Lotaria, que permite a uma pessoa sortuda aumentar o seu plafond, recebendo um prémio;
* 

*Figura 1-Menu de apresentação*

As funcionalidades extras acimas referidas serão explicadas de modo a facilitar o manuseamento do programa



*Figura 2-Menu das opções com as novas funcionalidades*



*Figura 3-menu das emergências*

3. 2. Estruturas utilizadas

Para o desenvolvimento deste programa utilizamos as seguintes estruturas:

* Lista Ligada de Pessoas (LLPessoas)
* Lista Ligada de Mesas (LLMesas)
* Lista Ligada de Refeições (LLRefeições)
* Nó da árvore que guarda as pessoas removidas (nóRemovidos)
* Lista Ligada de Inteiros para guardar números de departamento ou grupo (LLReserva)

**Na estrutura LLPessoas foram utilizados estes parâmetros:**

struct pessoa

{

struct aluno

{

int num; (número de aluno)

string curso; (curso do aluno)

bool especialOuNao; (indica se o aluno é especial, é true se tal for o caso)

};

struct staff

{

int numFuncionario; (número de funcionário)

};

string priNome; (é o primeiro nome da pessoa)

string ultNome; (é o último nome da pessoa)

int numDepartamentoOuGrupo; (é o número do grupo ou departamento)

int tamanhoGrupo; (é o tamanho do grupo ao qual a pessoa pertence)

float plafond; (é o plafond do individuo)

aluno membro\_aluno; (facilita a manipulação dos dados de um aluno)

staff membro\_staff; (facilita a manipulação dos dados de um funcionário)

int duração; (é a duração da refeição do indivíduo)

pessoa\*seguinte; (aponta para a pessoa seguinte)

};

pessoa\*primeira; (aponta para a primeira pessoa da lista)

pessoa\*iterator; (vai percorrer a lista quando for preciso)

pessoa\*ultima; (aponta para a última pessoa da lista)

**Na estrutura LLMesas foram utilizados estes parâmetros:**

struct mesa

{

int numMesa; (número de mesa)

int capacidade; (tamanho da mesa)

LLPessoas\* sentados; (aponta para a lista ligada de pessoas que representa as pessoas sentadas)

int numSentados; (número de pessoas sentadas na mesa)

mesa\*seguinte; (aponta para a mesa seguinte)

};

mesa\*primeira; (aponta para a primeira mesa da lista)

mesa\*iterator; (vai percorrer a lista quando for preciso)

mesa\*ultima; (aponta para a última mesa da lista)

**Na estrutura LLRefeições foram utilizados estes parâmetros:**

struct refeição

{

string entrada;

string pratoMain;

float custo;

refeição\*seguinte;

};

refeição\*atual; (aponta para a última refeição da lista, a que está disponível na cantina)

refeição\*primeira; (aponta para a primeira refeição da lista)

**Na estrutura nóRemovidos foram utilizados estes parâmetros:**

LLPessoas::pessoa\* removida; (aponta para a pessoa que foi removida)

nóRemovidos\*esquerda; (aponta para a sub-árvore esquerda)

nóRemovidos\*direita; (aponta para a sub-árvore direita)

**Na estrutura LLReserva foram utilizados estes parâmetros:**

struct item {

int num; (é o número de departamento ou grupo)

item\*seguinte; (aponta para o número seguinte)

};

item\*primeiro; (aponta para o primeiro número da lista)

item\*ultimo; (aponta para o último número da lista)

1. Utilização da Aplicação

4. 1. Menu Inicial

4. 2. Menu de Opções

void guardaFicheiros(string\* Lista, string nome);

int numLinhas(string nome\_ficheiro);

void limpaLLPessoas(LLPessoas\*ll);

void gravaPessoaRemovidos(LLPessoas::pessoa \* p);

void gravaLLPessoas(LLPessoas \* ll, const char\*NomeDoFicheiro);

void gravaRefeicoes(LLRefeições \* ref);

void gravaCantina(LLMesas\* cantina);

void gravaRemovidos(nóRemovidos\* raiz);

void carregaPessoa(string PriNome, string UltNome, string AlunoOuNao, string DepGru, string Curso, string numAluFunc, string duracao, string plaf, string tamGrupo, LLPessoas::pessoa\*p, LLReserva\*reserva);

void carregaLLPessoas(LLPessoas\* fila, LLReserva\*reserva, const char\*NomeDoFicheiro);

nóRemovidos\* carregaRemovidos(nóRemovidos\* arvoreRemovidos, LLReserva\*reserva, const char\*NomeDoFicheiro);

void limpaCantina(LLMesas\*cantina);

void carregaCantina(LLMesas\* cantina, int numMesas, LLReserva\*reserva, const char\*NomeDoFicheiro);

void limpaRefeições(LLRefeições\*r);

void carregaRefeições(LLRefeições \* r, const char\*NomeDoFicheiro);

int contaCantina(const char\*NomeDoFicheiro);

void novaMeal(LLRefeições \*pratos);

bool listaVaziaRefeições(LLRefeições\*ll);

void insereFimRefeições(LLRefeições\* ll, LLRefeições::refeição \*r);

void escreveMeal(LLRefeições\*ll);

void escreveTodasRefeições(LLRefeições\*ll);

bool listaVaziaCantina(LLMesas\*ll);

void insereFimMesas(LLMesas\* ll, LLMesas::mesa \*m);

void criaCantina(LLMesas\*ll);

void escreveCantina(LLMesas\*ll);

void preencheCantina(LLMesas \*cantina, LLPessoas\*fila, LLRefeições\*r);

void removeDuração(LLMesas\*cantina);

void retiraEmergPessoa(LLMesas\*cantina, LLPessoas\*acabados, LLRefeições \*r);

void retiraEmergGrupo(LLMesas\*cantina, LLPessoas\*acabados, LLRefeições \*r);

void removeAcabados(LLMesas\*cantina, LLPessoas\* acabados, LLRefeições\*r);

void insereFimPessoasAcabadas(LLPessoas\* ll, LLPessoas::pessoa \*p, LLRefeições\*r);

void ordenaAlfabeticamenteUltNome(LLMesas\*cantina, LLPessoas\*fila, LLPessoas\*acabados);

int contaPessoasCantina(LLMesas\*cantina);

void vagasPossiveis(int\*vagasPossiveis, LLPessoas::pessoa\*p, LLMesas\*cantina, int vagas);

void mudaNome(LLMesas\*cantina, LLPessoas\*fila);

void editaDuração(LLMesas\* cantina, LLPessoas\*fila);

LLMesas::mesa\*\* converteLLvetorMesas(LLMesas\*cantina);

void SelectionSortMesasNumSentados(LLMesas::mesa\*\*cantina, int tam);

void apresentaIndividuo(LLMesas\*cantina, LLPessoas\*fila);

void apresentaCursoOuDep(LLMesas\*cantina, LLPessoas\*fila, string \*CURSOS);

void mudaCurso(LLMesas\*cantina, LLPessoas\*fila, string\* CURSOS);

void PobreRicoEMedia(LLMesas\*cantina, LLPessoas\*fila);

void lotaria(LLMesas\*cantina, LLPessoas\*fila, bool \*háVencedor, int\*premio, int\*aumenta);

int comprimento(LLPessoas\* ll);

void limpaReserva(LLReserva\*ll);

bool listaVaziaPessoas(LLPessoas\*ll);

void inserirInicioPessoas(LLPessoas\* ll, LLPessoas::pessoa \*p);

void insereFimPessoas(LLPessoas\* ll, LLPessoas::pessoa \*p);

void insereMeioPessoas(LLPessoas\*ll, LLPessoas::pessoa\*p, int pos);

LLPessoas::pessoa\* removePessoaInicio(LLPessoas\*fila);

LLPessoas::pessoa\* removePessoaFim(LLPessoas\*fila);

LLPessoas::pessoa\* removePessoaMeio(LLPessoas\*fila, int pos);

LLPessoas::pessoa\* consultaPessoa(LLPessoas\*fila, int pos);

LLPessoas::pessoa\* criaPessoa(string pnome, string unome, string curso, int dura, int idOuDepart, float plafond, bool alunoOuNao, bool especialOuNao);

void preencheFila(LLPessoas\*fila, string\* pnomes, string\*unomes, string\*cursos, LLReserva\*reserva, bool primeiraVez);

void escreveFila(LLPessoas\* fila);

bool listaVaziaReserva(LLReserva\*ll);

void insereFimReserva(LLReserva\* ll, int num);

void testaGrupoDep(int\*num, LLReserva\*reserva);

nóRemovidos \* novoNó(LLPessoas::pessoa\*p);

nóRemovidos\* adicionaArvoreRemovidos(nóRemovidos\*raiz, LLPessoas::pessoa\*p);

int contaNosArvore(nóRemovidos\*raiz);

void guardaArvoreVetor(nóRemovidos \* raiz, int \*pos, LLPessoas::pessoa\*\*guardar);

void limpaArvore(nóRemovidos \* raiz);

void imprimeArvoreInfixa(nóRemovidos \* raiz);

nóRemovidos\* removeSemDinheiro(LLPessoas\*fila, LLRefeições \*r, nóRemovidos\*arvoreRemovidos);

nóRemovidos\* removeSemDinheiroPessoa(LLPessoas\*fila, nóRemovidos\*arvoreRemovidos, int numDepGrupo, int posiPessoa);

nóRemovidos\* removeSemDinheiroGrupo(LLPessoas\*fila, nóRemovidos\*arvoreRemovidos, int numDepGrupo);

void alterarPlafond(LLPessoas\*fila);

int contaPessoasFila(LLPessoas\*fila);

int contaAcabados(LLPessoas\*acabados);

void mergeSortAlfabeticamenteUltNome(LLPessoas::pessoa\*\*sistema, int tam);

void mergeUltNome(LLPessoas::pessoa\*\*left, LLPessoas::pessoa\*\*right, LLPessoas::pessoa\*\*sistema, int n\_left, int n\_right, int tam);