

Estrutura de Dados e 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

1. 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.

2. 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

3. 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 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;
- Possibilidade de ter banda sonora. *

Estas novas funcionalidades serão explicadas mais tarde neste relatório, de modo a não surgir dúvidas com a utilização do programa, exceto as que têm asterisco, que são fáceis de entender.

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)
```

4. Utilização da Aplicação

4.1. Menu Principal

Como o nome indica, é o menu principal do projeto, onde não foi efetuadas muitas alterações para facilitar a avaliação do mesmo.

```
Cantina EDA
Escolha uma opcao:
(s) Seguinte (e) Emergência (g) Gravar dados (c) Carregar dados (o) Opcoes

**** Comando: s

Refeição atual:
Entrada: Canja de Galinha
Prato Principal: Frango com arroz
Custo:3 EUR

Mesa 1(CAPACIDADE 5):
Baptista, Estudante, Grupo 309, Ciências da Educação, 2031713 (ciclos restantes: 2)
Rocha, Estudante, Grupo 309, Ciências da Educação, 2029112 (ciclos restantes: 2)
Rocha, Estudante, Grupo 309, Ciências da Educação, 2032117 (ciclos restantes: 2)
Oliveira, Estudante, Grupo 309, Ciências da Educação, 2037617 (ciclos restantes: 2)
Araújo, Staff, Departamento 111, 1999881 (ciclos restantes: 4)
-----
Mesa 2(CAPACIDADE 2):
Monteiro, Staff, Departamento 111, 1924490 (ciclos restantes: 4)
Pinheiro, Staff, Departamento 111, 1980273 (ciclos restantes: 4)
-----
Mesa 3(CAPACIDADE 3):
Torres, Staff, Departamento 111, 1974300 (ciclos restantes: 4)
Pires, Staff, Departamento 111, 1927477 (ciclos restantes: 4)
Gonçalves, Staff, Departamento 111, 1982891 (ciclos restantes: 4)
-----
Mesa 4(CAPACIDADE 4):
Martins, Staff, Departamento 103, 1937379 (ciclos restantes: 2)
Sá, Staff, Departamento 111, 1946875 (ciclos restantes: 4)
```

Figura 1 Exibição do menu principal.

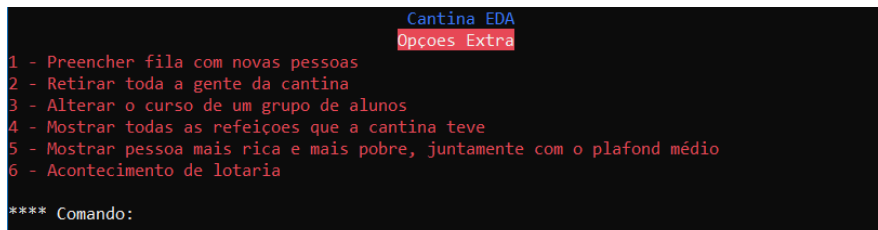
4.2. Carregamento de dados por parâmetro

De forma a assegurar o correto funcionamento da cantina, se pretender carregar os dados desta passando eles como parâmetros de execução, a passagem dos ficheiros deve seguir a seguinte ordem:

1. Refeicoes.txt
2. Cantina.txt
3. Fila.txt
4. Acabados.txt
5. Removidos.txt

4.3. Menu de Opções Extra

Ao pressionar a tecla 9, no menu das opções, intitulada “Opções Extra”, este menu aparece, com as 6 funcionalidades extra que implementámos.



```
Cantina EDA
Opções Extra
1 - Preencher fila com novas pessoas
2 - Retirar toda a gente da cantina
3 - Alterar o curso de um grupo de alunos
4 - Mostrar todas as refeições que a cantina teve
5 - Mostrar pessoa mais rica e mais pobre, juntamente com o placard médio
6 - Acontecimento de lotaria
**** Comando:
```

Figura 2 Menu das opções extra.

4.4. Preencher Fila

A tecla 1 do menu de opções extra permite apagar toda a gente que está na fila, para preencher a mesma com pessoas novas e o utilizador tem a opção de preencher com 10 grupos de pessoas ou só uma, como se pode ver abaixo.

```
Quer encher a fila com 10 grupos ou só 1?  
1.10 grupos;  
2.Só 1 grupo.  
**** Comando:
```

Figura 3 Função Preencher Fila.

4.5. Alterar curso

Ao pressionar a tecla 3 no menu de opções extra, é pedido um número de grupo de alunos ao utilizador, e se este grupo encontrar-se na fila ou na cantina, aparece o seguinte menu, para mudar o curso de todos os alunos desse grupo para o curso que o utilizador escolha.

```
Pretende mudar o curso de qual grupo de alunos?  
**** Comando: 226  
Cursos possíveis:  
1. Economia  
2. Educação Básica  
3. Matemática  
4. Línguas e Relações Empresarias  
5. Gestão  
6. Engenharia Informática  
7. Engenharia Electrónica e Telecomunicações  
8. Educação Física e Desporto  
9. Engenharia Civil  
10. Enfermagem  
11. Direcção e Gestão Hoteleira  
12. Artes Visuais  
13. Biologia  
14. Bioquímica  
15. Ciências da Educação  
16. Comunicação, Cultura e Organizações  
17. Design  
18. Psicologia  
19. Estudos de Cultura  
Indique o novo curso que pretenda que o grupo tenha.  
**** Comando: 
```

Figura 4 Função Altera Curso.

4.6. Rico, pobre e plafond médio

Com a tecla 5, o utilizador pode saber a pessoa com o maior, e menor, plafond na cantina e fila e o plafond médio.

```
Cantina EDA
A pessoa mais rica é o funcionário Eduarda Melo de número 1994593 que tem uns impressionantes 94.88 euros!
A pessoa mais pobre é o funcionário Mariana Ferreira de número 1919879 que tem uns miseráveis 3.51 euros.
O plafond médio é: 50.5595EUR.
Press any key to continue . . .
```

Figura 5 Função Rico, Pobre e Plafond Médio.

4.7. Lotaria

Ao pressionar a tecla 6, uma lotaria acontece para as pessoas na cantina e fila. Se houve vencedor da última vez que houve lotaria, ou se é a primeira vez que se faz a lotaria, é gerado um prémio aleatório entre 200 e 500 euros. Se ninguém tem a sorte de ganhar, o prémio duplica e a probabilidade de alguém ganhar o próximo prémio diminui. O prémio é acrescentado ao plafond da pessoa vencedora.

```
Cantina EDA
Opcoes Extra
1 - Preencher fila com novas pessoas
2 - Retirar toda a gente da cantina
3 - Alterar o curso de um grupo de alunos
4 - Mostrar todas as refeicoes que a cantina teve
5 - Mostrar pessoa mais rica e mais pobre, juntamente com o plafond médio
6 - Acontecimento de lotaria

**** Comando: 6
O vencedor da lotaria é o funcionário Vitória Raposo de número 1923779, que ganhou 306 euros! Parabéns!
Press any key to continue . . .
```

Figura 6 Função Lotaria.

5. Funcionalidades

Aqui será explicado o objetivo de cada função/grupo de funções.

- **As seguintes funções guardam os dados gerados pelo programa.**

- ✓ `void gravaLLPessoas(LLPessoas * ll, const char*NomeDoFicheiro);`
- ✓ `void gravaRefeicoes(LLRefeições * ref);`
- ✓ `void gravaCantina(LLMesas* cantina);`
- ✓ `void gravaRemovidos(nóRemovidos* raiz);`

- **As seguintes funções carregam os dados, anteriormente guardados pelas funções referidas acima.**

- ✓ `void carregaLLPessoas(LLPessoas* fila, LLReserva*reserva, const char*NomeDoFicheiro);`
- ✓ `nóRemovidos* carregaRemovidos(nóRemovidos* arvoreRemovidos, LLReserva*reserva, const char*NomeDoFicheiro);`
- ✓ `void carregaCantina(LLMesas* cantina, int numMesas, LLReserva*reserva, const char*NomeDoFicheiro);`
- ✓ `void carregaRefeições(LLRefeições * r, const char*NomeDoFicheiro);`

- **As seguintes funções limpam as listas ligadas, apagando o que continham para assegurar que os dados carregados não são adicionados a outros dados.**

- ✓ `void limpaCantina(LLMesas*cantina);`
- ✓ `void limpaRefeições(LLRefeições*r);`
- ✓ `void limpaReserva(LLReserva*ll);`
- ✓ `void limpaLLPessoas(LLPessoas*ll);`
- ✓ `void limpaArvore(nóRemovidos * raiz);`

- **As seguintes funções estão relacionadas com a manipulação das listas ligadas.**

- ✓ `bool` listaVaziaRefeições(`LLRefeições*ll`);
- ✓ `void` insereFimRefeições(`LLRefeições* ll`, `LLRefeições::refeição *r`);
- ✓ `bool` listaVaziaCantina(`LLMesas*ll`);
- ✓ `void` insereFimMesas(`LLMesas* ll`, `LLMesas::mesa *m`);
- ✓ `void` insereFimPessoasAcabadas(`LLPessoas* ll`, `LLPessoas::pessoa *p`, `LLRefeições*r`);
- ✓ `int` comprimento(`LLPessoas* 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`);
- ✓ `bool` listaVaziaReserva(`LLReserva*ll`);
- ✓ `void` insereFimReserva(`LLReserva* ll`, `int num`);

- **As seguintes funções têm o papel de criar a cantina, imprimi-la, inserir pessoas da fila na cantina (se possível), remover a duração das pessoas a comer e remover os acabados, cobrando-lhes a refeição, respetivamente.**

- ✓ `void` criaCantina(`LLMesas*ll`);
- ✓ `void` escreveCantina(`LLMesas*ll`);
- ✓ `void` preencheCantina(`LLMesas *cantina`, `LLPessoas*fila`, `LLRefeições*r`);
- ✓ `void` removeDuração(`LLMesas*cantina`);
- ✓ `void` removeAcabados(`LLMesas*cantina`, `LLPessoas* acabados`, `LLRefeições*r`);

- **As seguintes funções têm o papel de criar uma pessoa, preencher a fila com grupos de pessoas e imprimir a fila, respetivamente.**

✓ `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);`

- **As seguintes funções estão relacionadas com a situação de emergência.**

✓ `void retiraEmergPessoa(LLMesas*cantina, LLPessoas*acabados, LLRefeições *r);`
 ✓ `void retiraEmergGrupo(LLMesas*cantina, LLPessoas*acabados, LLRefeições *r);`

- **As seguintes funções têm o papel de receber uma nova refeição, escrever a refeição atual da cantina e todas as refeições que a cantina teve, respetivamente.**

✓ `void novaMeal(LLRefeições *pratos);`
 ✓ `void escreveMeal(LLRefeições*ll);`
 ✓ `void escreveTodasRefeições(LLRefeições*ll);`

- **As seguintes funções mostram as pessoas na fila, cantina e que acabaram ordenadas alfabeticamente pelo último nome, junto com o mergeSort que faz a ordenação; mostram as ordenadas pelo número de sentados através de selectionSort.**

✓ `void ordenaAlfabeticamenteUltNome(LLMesas*cantina, LLPessoas*fila, 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);`
 ✓ `void SelectionSortMesasNumSentados(LLMesas::mesa**cantina, int tam);`

- As seguintes funções mostram um indivíduo escolhido pelo utilizador, mostram um curso ou departamento da escolha do utilizador e mostram a pessoa com maior plafond, menor plafond e plafond médio da cantina e fila, respetivamente.

- ✓ `void apresentaIndividuo(LLMesas*cantina, LLPessoas*fila);`
- ✓ `void apresentaCursoOuDep(LLMesas*cantina, LLPessoas*fila, string *CURSOS);`
- ✓ `void PobreRicoEMedia(LLMesas*cantina, LLPessoas*fila);`

- As seguintes funções são responsáveis por remover as pessoas que não têm dinheiro para a refeição da fila.

- ✓ `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);`

- As seguintes funções estão relacionadas com o funcionamento da árvore da pesquisa binária, sendo a última responsável por mostrar os indivíduos removidos ordenados alfabeticamente pelo primeiro nome.

```
nóRemovidos * novoNó(LLPessoas::pessoa*p);
nóRemovidos* adicionaArvoreRemovidos(nóRemovidos*raiz, LLPessoas::pessoa*p);
int contaNosArvore(nóRemovidos*raiz);
void imprimeArvoreInfixa(nóRemovidos * raiz);
```

- As seguintes funções têm o papel de mudar o nome de um indivíduo, alterar a duração de um grupo/departamento, mudar o curso de um grupo de alunos, alterar o plafond de uma pessoa na fila e simular a lotaria, respetivamente.

- ✓ `void mudaNome(LLMesas*cantina, LLPessoas*fila);`
- ✓ `void editaDuração(LLMesas* cantina, LLPessoas*fila);`
- ✓ `void mudaCurso(LLMesas*cantina, LLPessoas*fila, string* CURSOS);`
- ✓ `void alterarPlafond(LLPessoas*fila);`
- ✓ `void lotaria(LLMesas*cantina, LLPessoas*fila, bool *háVencedor, int*premio, int*aumenta);`

- As restantes funções são funções auxiliares às funções já descritas.

6. Vantagens/desvantagens da solução

Vantagens:

- ✓ O utilizador tem acesso a todas as funcionalidades básicas da cantina e todos os dados desta.
- ✓ O programa não deve levantar dúvidas no seu funcionamento.
- ✓ O utilizador tem a escolha de várias opções e tem a possibilidade de manipular vários dados nos indivíduos presentes no programa.
- ✓ O programa cativa o seu utilizador ainda mais graças à sua estética.

Desvantagens:

- ✓ A quantidade de código implementado é significativa.

7. Conclusão e trabalho futuro

Com a elaboração deste projeto, os nossos conhecimentos sobre a linguagem C++ não só aprofundaram/melhoraram como também nos permitiu obter uma visão mais ampla e autónoma de como trabalhar com o mesmo. Apesar da complexidade de linguagem de C++ com a cooperação de todos foi possível trabalhar de forma eficiente.

A maior dificuldade deparada neste projeto foi a capacidade de assegurar as restrições de curso e de vagas quando entravam pessoas na cantina. Superadas as dificuldades com a ajuda dos docentes concluímos este projeto e agora estamos receptivos a novos desafios quer nesta quer noutra linguagem de modo a enriquecer a nossa capacidade de programação.

8. Referências

<https://stackoverflow.com/>

<http://moodle.cee.uma.pt/>

<http://www.cplusplus.com/>