Estruturas de Dados e Algoritmos

Projeto prático 2018/2019

(30% da avaliação da UC)

1. Objetivos

O objetivo do projeto é o desenvolvimento de um programa em C++ que simule o funcionamento da "**Cantina EDA**". O sistema deverá implementar/simular todas as funcionalidades relativas ao funcionamento de uma cantina. De uma forma sumarizada espera-se que o programa implemente funcionalidades para:

- Organização por grupos
- Fila de entrada na cantina
- Criação de refeições
- Faturação
- Movimentação de utilizadores da cantina

2. Descrição

As principais entidades do programa são a fila de espera, mesa, aluno, refeição e staff.

A **Fila de entrada** é representada por o número de alunos presente, e pelo conjunto de alunos presentes.

A entidade **mesa** representa uma mesa na cantina, o tamanho da mesa varia entre os 2 e os 5 lugares, cada mesa tem um número de alunos associado e um número de mesa. A cada inicialização do programa são criadas um conjunto de mesas com tamanhos aleatórios até preencher a capacidade máxima da cantina.

O **aluno** representa um aluno da universidade, e é representado pelo primeiro e último nome, número de aluno, número de grupo, curso e plafond. A entidade **staff** é identificada pelo primeiro e último nome, número de funcionário, pelo número do departamento a que pertence e plafond.

A entidade **refeição** (ou menu) representa as refeições na **cantina EDA** cada refeição tem uma entrada, prato principal e custo.

3. Funcionamento

3.1 Funcionamento geral

A simulação da cantina desenvolve-se por ciclos desencadeados pelo utilizador. Cada vez que o utilizador pressionar a tecla '**s' + 'enter'** um novo ciclo começa. Em cada ciclo o sistema deve (por esta ordem):

a. Remover os alunos e staff que já finalizaram a refeição
b. Cobrar a refeição a alunos e staff na saída
c. Mover alunos/staff da fila de espera para cantina (caso existam)
d. Gerar novos alunos/staff para colocar na fila
e. Na primeira iteração, deverá ser criada uma refeição. A cada 10 ciclos as refeições devem ser alteradas e neste caso é pedido ao utilizador para introduzir os dados necessários para as novas refeições.

3.1.1 Inicialização

De forma a facilitar o funcionamento do programa, assumimos que antes de qualquer aluno/staff entrar na cantina a fila está completa. Para a primeira entrega do projeto assumimos uma fila com capacidade máxima de 50 pessoas, enquanto que para a segunda entrega esta fila não deverá ter limite máximo. Só poderão ser descartados indivíduos se a fila já estiver completa.

3.1.1.1 Tamanhos da cantina e mesas

Quando o programa é inicializado, o tamanho da cantina é calculado aleatoriamente, a dimensão da cantina deverá variar entre os 30 e os 50 lugares. Após este cálculo, as mesas serão geradas aleatoriamente até preencherem os lugares disponíveis (tendo em conta a restrição apresentada em **2.**). A dimensão máxima calculada deverá obrigatoriamente ser respeitada pelas vagas nas mesas.

3.1.2 Operação

As refeições duram em média entre 2 a 5 ciclos, logo, sempre que um aluno/grupo/staff é criado um valor aleatório da duração da refeição que deverá ser respeitado.

Os alunos podem também pertencer a grupos, sendo que o grupo tem no mínimo 1 aluno (aluno individual) e no máximo 10 alunos. Alunos do mesmo grupo quando criados devem ficar em posições consecutivas da fila e partilham todos a mesma duração da refeição. Números de grupo ou de departamento não podem ser repetidos dentro da mesma simulação.

Alunos do mesmo grupo devem ser colocados na mesma mesa, caso o número de elementos do grupo seja superior à capacidade da mesa então o grupo deverá ser estendido para outra mesa, se não existirem mesas com vagas suficientes para colocar o grupo então esse grupo mantém-se na fila e outro grupo fica com a sua posição. Esta restrição mantém-se para membros do staff, no entanto neste caso os indivíduos são organizados de acordo com o departamento.

Alunos de cursos diferentes **não podem em nenhuma situação** partilhar a mesma mesa. Isto significa que poderão existir várias situações em que mesas ficam com lugares vazios pelo menos até chegar mais alguém do mesmo curso ou alguém do staff.

3.1.2 Persistência dos dados

A qualquer momento entre os ciclos o operador do programa pode pressionar a tecla **'g'** + **'enter'** para gravar todos os dados da cantina.

Cada grupo é livre de implementar o seu método e formato de gravação.

3.1.2.2 Carregamento de dados

No entanto, o sistema deverá permitir o carregamento dos dados se o(s) ficheiro(s) com os dados da cantina forem passados como parâmetros da execução do programa ou se o utilizador pressionar a tecla '**c**' + '**enter'** entre os ciclos.

3.2 Situações de emergência

Podem existir situações em que:

3.2.1 Um aluno/staff tem de sair da cantina a meio de uma refeição,

3.2.2 Um grupo/staff tem de sair da cantina a meio de uma refeição.

Em ambos os casos o sistema deverá pedir o número de aluno (ou grupo) ao utilizador e remover os elementos correspondentes da cantina (a refeição é na mesma cobrada). Esta operação é similar para os membros do staff, neste caso identificados pelo seu número de funcionário/departamento. Logo que este processo

aconteça novos indivíduos são inseridos na cantina para preencher as vagas criadas (não devemos esperar pelo próximo ciclo).

Estas situações de emergência ficam disponíveis para o utilizador do programa após pressionar a tecla '**e**' + '**ente**r'.

3.2.3 Alunos com estatuto especial têm o direito de passar diretamente para a frente da fila, nesta simulação deverá haver uma probabilidade de 5% de cada aluno criado ter estatuto especial. Este tipo de aluno (por exemplo trabalhadores estudantes) para efeitos de refeição não pertencem a nenhum grupo (grupo individual).

3.3 Criação de alunos e membros do staff

A geração de alunos, staff deverá ser feita **de forma aleatória** a cada ciclo.

O primeiro e segundo nome destas entidades deverá ser retirado aleatoriamente dos ficheiros primeiro_nome.txt e segundo_nome.txt, o número de aluno e de funcionário deverão ser identificadores únicos no programa e serão usados para os identificar durante e execução do programa. O grupo e id do departamento também deverá ser um valor único (partilhado entre os membros de um grupo/departamento) entre 100 e 500. O curso do aluno deverá ser retirado aleatoriamente dos ficheiros cursos.txt. O plafond de cada indivíduo deverá ser um valor aleatório entre 1 e 100 euros.

3.3.1 Ficheiros

Os ficheiros *primeiro_nome.txt, segundo_nome.txt, cursos.txt*, serão iguais para todos os grupos e estão disponíveis aqui.

3.4 Fila de Espera

A fila de espera da cantina funciona como uma fila "normal", ou seja, os primeiros a chegar serão os primeiros indivíduos a entrar, e quem chega por último é colocado nas últimas posições da Fila. **Salvo os indivíduos com o estatuto especial mencionado acima.**

3.4.1 Restrições entrada

Caso um indivíduo não tenha plafond suficiente para pagar a refeição este deverá ser removido antes de entrar na cantina, nestes casos deverá ser dada a oportunidade ao utilizador do programa de remover apenas esse aluno/staff ou o grupo/departamento inteiro. Todos os alunos/staff removidos deverão ficar guardados no programa, para a primeira fase do projeto assume-se que no máximo serão guardados 100 elementos, enquanto que para a segunda fase espera-se que estes elementos sejam guardados numa árvore de pesquisa binária de acordo com o seu primeiro nome.

4 Visualização

A visualização do programa na consola deverá seguir o seguinte formato (repetido a cada ciclo)

```
                        Cantina EDA
(s)Seguinte (e)Emergência (g)Gravar (c)Carregar Dados (o)Opções

Refeição actual:
      Entrada: Canja de galinha
      Prato: Jardineira
      Custo: 4,5€

MESA 1 (CAPACIDADE 3):
      Quintal , estudante , LEI, Grupo 1, 74612 (ciclos restantes : 4)
      Alves, estudante, LEI, Grupo 1, 84716 (ciclos restantes : 4)

MESA 2 (CAPACIDADE 5):
      Martin, estudante, CCO, Grupo 14, 91802 (ciclos restantes : 1)
      Simon, estudante, CCO, Grupo 14, 81721 (ciclos restantes : 1)
      Jesus, estudante, CCO, Grupo 11, 91827 (ciclos restantes : 3)
      Alves, estudante, CCO, Grupo 11, 81234 (ciclos restantes : 3)
      Pereira, estudante, CCO, Grupo 11, 88178 (ciclos restantes : 3)

MESA 3 (CAPACIDADE 4):
      Silva, estudante CCO, Grupo 11, 99182 (ciclos restantes : 3)

MESA 4 (CAPACIDADE 5):
      Caldeira, staff, Departamento 1, 88172 (ciclos restantes : 5)
      Clarke, staff, Departamento 1, 88712 (ciclos restantes : 5)
      Wilson, staff, Departamento 1, 99123 (ciclos restantes : 5)
      Freitas, estudante, LEIRE, Grupo 9, 9912 (ciclos restantes : 2)
      Barreto, estudante, LEIRE, Grupo 9, 22124 (ciclos restantes : 2)

MESA 5 (CAPACIDADE 2):
      Mendes, estudante, LEIRE, Grupo 9, 22991 (ciclos restantes : 1)
      Faria, estudante, LEIRE, Grupo 14, 88172 (ciclos restantes : 1)
…
[Repetir para todas as Mesas]

FILA DE ESPERA:
Lavera, Estudante, Grupo 20, LE, Duração, 3, 13€
Un, Estudante,     Grupo 21, MEI, Duração, 4, 14€
Jacki, Estudante, Grupo 21, MEI, Duração, 4, 40€
Kelsi, Estudante, Grupo 21, MEI, Duração, 4, 11€
Elton, Estudante, Grupo 21, MEI, Duração, 4, 15.5€
Blair, Estudante, Grupo 25, LEC, Duração, 2, 12€
Ariane, Estudante, Grupo 25,   LEC, Duração, 2, 14.4€
Ervin, Staff, Departamento 7, Duração, 1, 10€
Brittni, Staff, Departamento 7 , Duração, 1, 10€
Trinidad, Staff, Departamento 7, Duração, 1, 12€
Harrison, Estudante, Grupo 30, CBM, Duração, 4, 11.7 €
Lindsay, Estudante, Grupo 30   , CBM, Duração, 4, 1€
Gavin, Estudante, Grupo 30,    CBM, Duração, 4, 1€
Ria, Estudante, Grupo 33 (especial), CBM, Duração, 4, 2€
Hildegarde, Estudante, Grupo 34, LEB, Duração, 5, 21.5€
Katlyn, Estudante, Grupo 34    LEB, Duração, 5, 25€
Ian, Staff, Departamento 8, Duração, 1, 22€
Miguelina, Staff, Departamento 9, Duração, 4, 13€

**** Commando:
```

Figura 1 : Exemplo de visualização, apresentação principal

```
***** ATENÇÃO ******
O aluno com o número 182731 não possui plafond suficiente para iniciar a
refeição
   1. Remover aluno da entrada
   2. Remover grupo da entrada
**** Commando:
```

Figura 2 : Exemplo de visualização, remoção de aluno ou grupo da fila de espera, (a apresentação para o staff/departamento) deverá ser similar

```
***** EMERGENCIA ******
Situação de emergência,
   3. Remover aluno/staff da cantina
   4. Remover grupo/departamento da cantina
**** Commando:
```

Figura 3: Exemplo de visualização, situação de emergência, após esta selecção o utilizador deverá escolher o número de aluno/staff ou grupo/departamento.

```
***** REFEIÇÃO NOVA ******
A cantina EDA necessita de uma nova refeição
   1. Introduza a entrada:
   2. Introduza o prato principal:
   3. Introduza o preço:
```

Figura 4: Exemplo de visualização, criação de uma nova refeição, os passos 1, 2 e 3 deverão ser intercalados com o input do utilizador

5 Outras funcionalidades

Entre cada ciclo o sistema deverá permitir as seguintes operações, estas deverão ser apresentadas num menu caso o operador pressione a tecla '**o**' (Opções):

5.1 Mostrar todos os indivíduos no sistema (primeiro, último nome, grupo e curso/departamento, plafond para verificar se a condição foi verificada)

5.1.1 Mostrar todos os indivíduos (primeiro, último nome, grupo e curso/departamento) ordenados por ordem alfabética do último nome

5.2 Mostrar todas as mesas (número e membros)

5.2.1 Mostrar todas as mesas (número e membros) ordenadas pelo número de lugares ocupados.

5.3 Mostrar todos os indivíduos que tiveram de ser rejeitados por falta de plafond (primeiro, último nome, grupo e curso/departamento)

5.3.1 Mostrar todos os indivíduos que tiveram de ser rejeitados por falta de plafond (primeiro, último nome, grupo e curso/departamento) ordenados pelo primeiro nome

5.4 Alterar o plafond de um indivíduo enquanto este ainda se encontra na Fila, através do seu id ou número de aluno

5.5 Pesquisa e apresentação sobre os indivíduos de um determinado curso/departamento

5.6 Editar a duração de uma refeição de um grupo/departamento (deverá afetar todos os elementos desse grupo)

5.7 Pesquisa sobre os indivíduos com base no número de funcionário/aluno. A apresentação deverá mostrar todos os detalhes do mesmo

5.8 Editar o nome de um indivíduo

6 Outras observações

Este enunciado poderá gerar alguns problemas de interpretação por parte dos alunos, desta forma todas as questões relativas à interpretação do enunciado deverão ser colocadas na ferramenta EDA overflow no site da disciplina.

Ao longo do projecto será necessário utilizar valores aleatórios diversas vezes, para isto é aconselhado que os alunos utilizem a função srand(seed) em que é passado um seed, normalmente a data actual em milisegundos. Após esta definição (idealmente no início do programa) deverá ser utilizada a função rand() para obter valores pseudo-aleatórios, exemplos aqui: http://www.cplusplus.com/reference/cstdlib/srand/ e https://www.programiz.com/cpp-programming/library-function/cstdlib/srand. A função srand deverá ser chamada apenas 1 vez no projecto.

É também claro que ao longo da execução do programa será necessário aceder a ficheiros diversas vezes para retirar nomes ou cursos aleatórios. De forma a diminuir o acesso aos mesmos, aconselha-se que os alunos criem arrays que são carregados com os conteúdos dos ficheiros no início da execução, depois o acesso deverá ser sempre feito a esses arrays.

7 Entregas

7.1

1ª Entrega 1/05/2019 às 23:59

- Especificação dos tipos de dados **(structs) necessários,** entrega de um documento com o máximo 2 páginas a explicar a especificação das structs.
- Implementação da lógica da cantina (descrita em **3. (e todas as subsecções)**
- Implementação da visualização do programa como definido em **4.**
- Implementação das funções descritas nos pontos, **5.1, 5.1.1 e 5.4**
- **Implementação utilizando vetores alocados dinamicamente.**

Apresentação/discussão do trabalho por todos os membros do grupo na semana seguinte à entrega (horário a definir).

2ª Entrega 1/06/2019 às 23:59

- **Implementação do projecto utilizando listas ligadas.**
- Implementação de todas as funcionalidades não implementadas na 1ª entrega.
- Implementação de funcionalidades extra (estas funcionalidades extra devem estar bem especificadas no relatório)
- Relatório (.pdf)

Apresentação/discussão do trabalho por todos os membros do grupo na semana seguinte à entrega (horário a definir)

8 Relatório

O relatório do projeto deve conter:

1. Introdução – breve descrição do objetivo do relatório
2. Problema – breve descrição do problema que estão a resolver
3. Solução proposta
    a. Descrição geral da solução
    b. Descrição de aspetos particulares relevantes (decisões da implementação que achem importantes de mencionar)
4. Utilização da aplicação - breve descrição de como utilizar a aplicação
5. Discussão – análise da solução proposta
    a. Vantagens / desvantagens da solução desenvolvida
6. Conclusão e trabalho futuro
7. Referências
8. Anexos

Avaliação

- Aplicação adequada do paradigma de programação imperativa;
- Definição de estruturas de dados adequadas;
- Cumprimento dos objetivos;
- Qualidade do código desenvolvido;
- Qualidade de execução do programa desenvolvido;
- Qualidade da interação com o utilizador (usabilidade);
- Relatório e documentação do código;
- Apresentações/discussões.

9 Código de ética e honestidade académica

Nesta disciplina, espera-se que cada aluno subscreva os mais altos padrões de honestidade académica. Isto significa que cada ideia que não seja do aluno deve ser explicitamente creditada ao(s) respetivo(s) autor(es). O não cumprimento do disposto constitui uma prática de plágio. O plágio inclui a utilização de ideias, código ou conjuntos de soluções de outros alunos ou indivíduos, ou qualquer outra fonte para além dos textos de apoio à disciplina, sem dar o respetivo crédito a essas fontes. A menção das fontes não altera a classificação, mas os alunos não deverão copiar código de outros colegas, ou dar o seu próprio código a outros colegas em qualquer circunstância. De notar que a responsabilidade de manter o acesso ao código somente para os colegas de grupo é de todos os elementos.