Estruturas de Dados e Algoritmos

Projeto prático 2 - 2019/2020

(15% da avaliação da UC)

1. Objetivos

O objetivo do projeto é o desenvolvimento de um programa em C++ que simule o funcionamento da empresa "**ExpressoEDA**". Esta é uma empresa de transporte de passageiros de ponto a ponto (não podem entrar passageiros nas paragens intermédias, apenas sair). O sistema deverá implementar/simular todas as funcionalidades relativas ao funcionamento desta empresa, de uma forma geral espera-se que seja simulada a:

- Entrada de passageiros na origem
- Saída de passageiros nos destinos

Espera-se que o projeto seja desenvolvido utilizando tipos de dados definidos pelos alunos (structs), listas ligadas e árvores de pesquisa binária

0123456789ABCDEF

A utilização da classe/biblioteca vector não é permitida.

1000 <-> FFFF -> converter para

Cada viagem é levada a cabo por um autocarro, cada autocarro terá uma capacidade de passageiros, um motorista, e uma matrícula.

O motorista é identificado pelo seu primeiro e último nome. A matrícula é composta por 4 valores hexadecimais aleatórios.

Cada autocarro tem também associada uma lista de passageiros, de forma a simplificar a operação assumimos que todos os autocarros iniciam a sua viagem com todas as vagas preenchidas.

2. Inicialização

Quando o programa inicializar deverão ser considerados os seguintes pontos:

- A paragem inicial e final deverão ser aleatórias
- O número de paragens intermédias também deverá ser um valor aleatório entre 4 e 9 (inclusivé)

2.1 Criação dos passageiros

Cada passageiro é identificado por um primeiro e último nome, e pelo número do bilhete (valor aleatório que não poderá ser repetido durante a execução do programa). O primeiro e último nome dos passageiros deverão ser retirados aleatoriamente do ficheiro disponível aqui.

Aquando da inicialização, o programa deverá criar 30 passageiros que serão colocados numa fila de espera, esta fila de espera deverá seguir um paradigma FIFO.

2.2 Criação dos autocarros

A cada criação de um autocarro os primeiros elementos da fila deverão ser removidos e associados ao autocarro, este número dependerá da capacidade do autocarro. A capacidade do autocarro é um valor aleatório entre 5 e 10.

2.1 Criação das paragens

Cada paragem de autocarro é identificada pelo seu nome e por todos os passageiros que terminaram a viagem nessa paragem (aqui identificados apenas pelo número do seu bilhete). Os nomes da paragens de autocarros não poderão se repetir durante a execução do programa, e deverão ser retirados aleatoriamente do ficheiro disponível aqui. De forma a facilitar a pesquisa cada paragem de autocarro possui uma árvore de pesquisa binária que contém os números de todos os bilhetes dos passageiros que terminaram aí a sua viagem.

2.2 Percurso e movimentação de autocarros

Espera-se que a movimentação de autocarros seja implementada utilizando uma lista ligada com capacidade máxima igual ao número de paragens. A cada ciclo de execução (ver secção 3), os autocarros movimentam-se em direcção à última paragem, após a última paragem estes serão removidos do sistema.

Na prática este movimento é análogo a retirar um elemento (autocarro) no final de uma lista e adicionar novo elemento no ínicio.

3. Funcionamento

O funcionamento deverá seguir uma simulação em que o input do utilizador será mínimo. Na primeira iteração não existem autocarros nas paragens, no entanto a fila de espera e primeiro autocarro poderão estar criados.

A partir deste momento o funcionamento do programa segue iterações iniciadas quando o utilizador pressionar s+enter. A cada iteração o programa deverá:

- Movimentar o autocarro para a paragem seguinte
- Remover o autocarro que partiu da última paragem
- Criar 15 novos passageiros e colocá-los na fila de espera
- Remover um conjunto aleatório de passageiros de cada autocarro (ver secção 3.1)
- Colocar o números dos bilhetes de cada passageiro na árvore de cada paragem
- Voltar a apresentar o estado do programa de acordo com a secção 4.

3.1 Saída de passageiros

A cada paragem existe uma probabilidade de 25% de um passageiro terminar aí a sua viagem. Esta probabilidade é igual para todos os passageiros. Assim, este valor deverá ser considerado para todos o passageiros de um determinado autocarro, e os passageiros em questão deverão ser removidos. É importante relembrar que os números de bilhete de cada passageiro que termine a sua viagem deverá ser adicionado à árvore de pesquisa binária da paragem em questão.

4. Visualização

O programa deverá seguir o modelo abaixo na sua apresentação

```
Fila de Espera:
 Silva                          Marques                        Neves
 Santos                         Alves                          Coelho
 Ferreira                       Almeida                        Cruz
 Pereira                        Ribeiro                        Cunha
 Oliveira                       Pinto                          Pires
 Costa                          Carvalho                       Ramos
 Rodrigues                      Teixeira                       Reis
 Martins                        Moreira                        Simões
 Jesus                          Correia                        Antunes
 Sousa                          Mendes                         Matos


Paragem : Ribeira Brava
Autocarro : MXD1 Motorista: Maria Amaral
Passageiros : Pedro 23441 , Santiago 12341, ,Valter 12314, Alice 1235, Benedita l112512 Carmo 77634,
Diana 23411
(REPETIR PARA TODAS AS PARAGENS)

(s)eguinte  (o)pcoes
|
```

Figura 1 : Exemplo de apresentação do programa

Na apresentação, a fila de espera deverá mostrar 3 colunas, em que elementos mais à esquerda serão os últimos a sair e os elementos mais à direita e mais abaixo serão os primeiros a ser colocados em autocarros (neste exemplo seriam os passageiros Matos e Antunes). Na fila de espera os passageiros são identificados pelo seu último nome.

Na paragem é apresentado o seu nome, o autocarro na paragem, seu motorista e passageiros (aqui identificados pelo primeiro nome e número do bilhete).

5. Operações

Ao pressionar **o+enter** as seguintes opções deverão ser apresentadas ao utilizador:

1. Remover passageiros nos autocarros
2. Remover passageiros em fila de espera
3. Apresentar bilhetes por paragem
4. Alterar motorista
5. Remover bilhete da paragem

Após qualquer uma destas operações o sistema deverá voltar a apresentar o estado do programa.

5. 1 Remover passageiros nos autocarros

Em situações especiais poderá ser necessário remover passageiros de um autocarro, para isso o sistema deverá pedir o número do bilhete a remover. Esta operação deverá acontecer após a saída dos passageiros em cada paragem.

5.2 Remover passageiros em fila de espera

Da mesma forma, deverá também ser possível remover passageiros na fila de espera, o procedimento deverá ser o mesmo, ou seja através do número do bilhete de cada passageiro

5.3 Apresentar bilhetes por paragem

Deverá ser possível ao utilizador visualizar todos os números do bilhetes dos passageiros que terminaram a viagem numa determinada paragem. Para isso o utilizador deverá especificar qual a paragem que pretende visualizar e o sistema deverá apresentar esta informação ordenada pelo número de bilhete **(5.3.1**), ou sob a forma de uma árvore de pesquisa binária (ver aula teórica 13 parte 1 função imprime árvore) **(5.3.2)**

5.4 Alterar motorista

Deverá ser possível ao utilizador alterar o nome do motorista, para isto o utilizador deverá especificar a matrícula do autocarro e o sistema deverá pedir o novo nome do motorista.

5.5 Remover bilhete da paragem

Deverá ser possível remover um determinado bilhete da lista (árvore) associada a uma certa paragem de autocarro. Esta remoção poderá seguir qualquer um dos métodos de remoção de elementos em árvores de pesquisa binária discutidos nas aulas.

6. Entrega

A entrega do projeto será finalizada no moodle, na data de entrega será criado um formulário no moodle em que cada grupo deverá submeter um ficheiro .zip ou .rar com o relatório e todo o código e ficheiro necessários para a execução/avaliação. O nome do ficheiro entregue deverá ser o mesmo do número do grupo, por exemplo Grupo1.zip.

A entrega deste projeto está agendada para o dia 13/06/2020 às 23:59:59.

Na semana seguinte à entrega serão agendadas reuniões com os membros de cada grupo de forma a aferir a contribuição de cada um para o projeto. Esta reunião pode influenciar a nota individual até 100%, na prática isto significa que no mesmo grupo poderão haver colegas avaliados em 18 enquanto que outros poderão ter uma avaliação abaixo da nota mínima (**o que implica a reprovação da disciplina**).

6.1 Critérios de 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.

7. Notas implementação

7.1 Criação de valores aleatórios

Durante a inicialização do programa existem várias variáveis que serão aleatórias.
Para implementar este comportamento espera-se a utilização da função **`srand`**, a utilização deverá seguir os seguintes passos:

- **No "main" do programa** : importar as bibliotecas **`<stdlib.h>`** e **`<time.h>`**
  - criar o "seed" para a função rand através da instrução **`srand (time(NULL));`**
- Para o cálculo de um número aleatório (em qualquer ficheiro)
  - importar o **`<stdlib.h>`**
  - Utilizar a função **`rand()%NÚMERO`** para calcular um valor aleatório entre 0 e um (**NÚMERO -1**) definido

```
/* valor aleatório entre 1 e 10: */
  valor = rand() % 10 + 1;
```

7.2 Apresentação formatada

Para a apresentação da fila de espera proposta em 4, grupos deverão usar a função setw de forma a garantir o alinhamento correto dos últimos nomes dos passageiros, mais informação disponível aqui http://www.cplusplus.com/reference/iomanip/setw/.

Existem também outras funções na bilbioteca iomainp que poderão ajudar os grupos na apresentação

Cada grupo é livre de implementar soluções para todos os restantes detalhes de implementação, não definidos neste enunciado.

8. 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.