Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Biológicas
Departamento de Computação

BCC701
Programação de Computadores I
2020-01, remoto

Questões práticas – Estruturas Homogêneas (Parte 2 - Vetores)

ATENÇÃO:

- Para esta atividade prática você deverá incorporar, sempre que necessário, as funções de manipulação de vetores apresentadas no *Livro Texto - Módulo 7 - Estruturas Homogêneas*, mais especificamente, Seções *7.3.1, 7.3.3 e 7.3.5*.
- Eventualmente, você deverá criar novas funções, ou modificar as funções apresentadas, para tratar os dados de uma maneira mais específica ou diferente, conforme definições do enunciado da questão.

Questão 1

Sua tia vende dois tipos de salgados, coxinha e quibe. Com o aumento dos pedidos, ela contrata vendedores temporários. Ela deseja premiar o vendedor que conseguir o maior lucro e dispensar o que conseguir o menor lucro. A fim de ajudá-la nesta tarefa, você vai implementar um programa que funciona da seguinte forma:

1. Faz a leitura de quantas unidades de coxinhas foram vendidas por cada vendedor, separadas por vírgula, em linha única. Em seguida, preenche-se um vetor com as quantidades lidas, utilizando-se a função `preencherVetor` (definida na Seção 7.3.3);
2. Realizar o mesmo procedimento anterior para o preenchimento das vendas de quibe. **Dica**: Os dois vetores estão associados pelos seus índices, ou seja, o índice 0 dos dois vetores armazena as vendas de cada salgado para o primeiro vendedor, o índice 1 armazena as vendas do segundo vendedor e assim por diante;
3. Valida os dois vetores, verificando se eles possuem a mesma quantidade de elementos. Caso inválido, imprime a mensagem "Dados de vendas inválidos". Caso contrário:
   (a) Lê dois valores reais, representando os valores dos lucros por unidade de coxinha e de quibe, respectivamente;
   (b) Preenche um terceiro vetor, contendo o valor arrecadado por cada vendedor. Para isso, você deve implementar a função `calculaLucros`, que recebe os dois vetores de vendas e os dois valores de lucro, e retorna um vetor contendo o lucro total de cada vendedor;
   (c) Imprime o vetor preenchido no item anterior utilizando a função `imprimeVetor` (definida na Seção 7.3.5), adaptada para imprimir os valores com duas casas decimais.
   (d) Determina e imprime o maior e o menor lucros obtidos pelos vendedores.

Siga o padrão dos exemplos mostrados abaixo.

Exemplo 1:

```
Vendas de coxinhas: 5, 2, 3, 10, 8
Vendas de quibes: 7, 3, 2, 8, 10, 5
Dados de vendas inválidos
```

Exemplo 2:

```
Vendas de coxinhas: 5, 2, 3, 10, 8
Vendas de quibes: 7, 3, 2, 8, 10
Lucro por unidade de coxinha: R$ 4.5
Lucro por unidade de quibe: R$ 2.8
Lucros: [ 42.10, 17.40, 19.10, 67.40, 64.00 ]
Maior lucro: R$ 67.40
Menor lucro: R$ 17.40
```

Exemplo 3:

```
Vendas de coxinhas: 6, 2, 10, 4, 7, 1
Vendas de quibes: 5, 4, 8, 10, 8, 2
Lucro por unidade de coxinha: R$ 3.5
Lucro por unidade de quibe: R$ 2.1
Lucros: [ 31.50, 15.40, 51.80, 35.00, 41.30, 7.70 ]
Maior lucro: R$ 51.80
Menor lucro: R$ 7.70
```

Questão 2

No enfrentamento à COVID-19, um importante parâmetro de avaliação é a "Redução (R)" ou "Aumento (A)" do número de mortes contabilizado(a) ao longo das semanas. Implemente um programa que lê o número de mortes de cada macro-região na última semana e na semana anterior. Estas informações são definidas em dois vetores de números inteiros, cuja entrada é feita a partir de um texto para cada vetor e o seu preenchimento com a função `preencherVetor` (definida na Seção 7.3.3). O programa deve verificar se os vetores possuem o mesmo número de elementos. Em caso afirmativo, ele processa os dados e gera o resultado descrito a seguir. Caso contrário, ele apenas imprime uma mensagem de erro, conforme os exemplos de execução apresentados posteriormente.

Os dois vetores preenchidos estão associados pelos seus índices e cada índice representa uma macro-região do estado. A semana anterior e a semana atual de cada macro-região são comparadas e um novo vetor é preenchido com as letras 'R', 'A' ou 'E', representando "Redução", "Aumento" ou "Estabilidade", respectivamente. Este novo vetor é obtido por uma função implementada por você, a `comparaVetores`, que recebe os dois vetores e retorna o novo vetor preenchido. Em seguida, o resultado é impresso no terminal utilizando a função `imprimeVetor` (definida na Seção 7.3.5). Por fim, o estado em análise é classificado como "Onda Vermelha" se a quantidade de "reduções" for menor do que a quantidade de "aumentos" no número de mortes, "Onda Verde" se for maior, e "Onda Amarela" se for igual. A classificação é obtida por uma função implementada por você, a `classificaEstado`, que recebe o vetor resultante das comparações e retorna a classificação do estado, conforme as ondas definidas anteriormente).

Siga o padrão dos exemplos mostrados abaixo.

Exemplo 1:

```
Número de mortes na semana anterior: 250, 300, 210, 15, 20
Número de mortes na semana atual: 200, 302
Número de elementos incompatível (5 != 2)
```

Exemplo 2:

```
Número de mortes na semana anterior: 250, 300, 210, 15, 20
Número de mortes na semana atual: 200, 302, 154, 3, 20
Classificações das macro-regiões:
[ R, A, R, R, E ]
Classificação do estado: Onda Verde
```

Exemplo 3:

```
Número de mortes na semana anterior: 250, 300, 210, 15, 20
Número de mortes na semana atual: 250, 300, 210, 15, 20
Classificações das macro-regiões:
[ E, E, E, E, E ]
Classificação do estado: Onda Amarela
```

Questão 3

A prefeitura te contratou para coordenar a vacinação de seus habitantes através de etapas baseadas em faixas etárias. Para realizar esta tarefa, você implementará um programa que contabilizará o número de vacinas necessárias em cada etapa. As etapas definidas são: (1) ≥ 85 anos; (2) < 85 e ≥ 65 anos; (3) < 65 e ≥ 45 anos; e (4) < 45 e ≥ 18 anos. Primeiro você precisa ler um vetor contendo as idades dos moradores cadastrados, a partir de uma string contendo todas as idades e depois convertendo a string para um vetor de inteiros, usando a função `preencherVetor` (definida na Seção 7.3.3). Em seguida, você contabiliza a quantidade de moradores em cada faixa etária. Esta tarefa será implementada na função `contabilizarDemandas`, que recebe o vetor de idades e retorna um novo vetor contendo a quantidade contabilizada para cada faixa etária (ou seja, cada uma das 4 faixas etárias representa um elemento deste vetor). Para iniciar este "vetor contador" com o seu valor padrão, você utilizará a função `criarVetor` (definida na Seção 7.3.1). As demandas de cada faixa etária, definidas no "vetor contador" obtido, são impressas no terminal conforme os exemplos de execução. Você também fará a leitura da quantidade de vacinas disponíveis para cada etapa, também em um vetor, lido inicialmente como string e convertido posteriormente para um vetor de inteiros. Em seguida, você avaliará se todas as faixas etárias poderão ser atendidas. Isso é feito através de comparações dos pares de demanda e disponibilidade representados pelos dois vetores correspondentes. Esta tarefa será implementada na função `avaliaAtendimento`, que recebe os dois vetores ("contador" e disponibilidade de vacinas) e retorna um valor Boolenano `True`, indicando que é possível realizar a vacinação de todas as faixas etárias, ou `False`, caso contrário. **Dica**: se pelo menos uma faixa etária não pode ser atendida, a resposta da função é `False`. Vacinas destinadas a uma etapa não podem ser usadas em outra.

Siga o padrão dos exemplos mostrados abaixo e considere que as entradas são sempre válidas.

Exemplo 1:

```
Defina as idades dos habitantes: 10, 15, 25, 35, 45, 55, 65, 85, 90
Demandas a serem atendidas por faixa etária:
. >= 85.........: 2
.  < 85 e >= 65.: 1
.  < 65 e >= 45.: 2
. >= 18.........: 2
Defina as disponibilidades de vacinas: 3, 1, 2, 4
A quantidade de vacinas é suficiente.
```

Exemplo 2:

```
Defina as idades dos habitantes: 1, 95, 20, 40, 50, 0, 60, 80, 30, 85
Demandas a serem atendidas por faixa etária:
. >= 85.........: 2
.  < 85 e >= 65.: 1
.  < 65 e >= 45.: 2
. >= 18.........: 3
Defina as disponibilidades de vacinas: 5, 1, 1, 1
Infelizmente, precisaremos de mais vacinas.
```