Programação de Computadores II [BCC702]

Segunda prova (2019–2)
23 de outubro de 2019
Matrícula: _____
Nome: _____

Departamento de Computação
Universidade Federal de Ouro Preto
Prof. José Romildo Malaquias

A organização e legibilidade do código serão avaliados.

Questão 1. [3 pontos]

Faça um programa para verificar o estoque de tecidos em uma loja de tecidos.

Os dados dos produtos em estoque devem ser obtidos de um **arquivo texto** `estoque.txt`. Considere que cada linha do arquivo contém os seguintes dados de um determinado item do estoque:

- código do item (*inteiro*)
- tipo do tecido (*string*)
- cor do tecido (*string*)
- metragem disponível (*double*)

Por exemplo:

```
439 tergal   azul      21.75
677 cambraia branco    13.4
701 linho    azul       9.5
702 viscose  amarelo   12.75
703 seda     marrom    21.3
855 chita    estampado 50
```

1. Defina um tipo registro para representar items do estoque.
2. Defina uma função *venda* que recebe como argumentos um vetor de ítens, a quantidade ítens do vetor, e o código e a metragem do item que o cliente deseja comprar. A função deve verificar se há tecido suficiente para a venda, de acordo com a metragem, retornando *true* se o código estiver cadastrado e houver quantidade suficiente do tecido para a venda, e *false* em caso contrário.
3. Defina a função *main* que
   (a) cria um vetor com os dados do estoque disponível na loja obtidos do arquivo `estoque.txt`, considerando que a quantidade máxima de ítens em estoque é 1000;
   (b) lê da entrada padrão o código e a metragem do tecido que o cliente deseja comprar;

(c) use a função *venda* para verificar a disponibilidade do item que o cliente deseja comprar e exibe uma das mensagens `Venda realizada com sucesso!`, ou `Não foi possível realizar a venda!`, de acordo com o resultado da verificação.

Exemplos de execução do programa:

```
Loja de Tecidos Bela Estampa
Digite o código do tecido: 333
Digite a metragem desejada: 5.6
Não foi possível realizar a venda!
```

```
Loja de Tecidos Bela Estampa
Digite o código do tecido: 701
Digite a metragem desejada: 21.3
Não foi possível realizar a venda!
```

```
Loja de Tecidos Bela Estampa
Digite o código do tecido: 701
Digite a metragem desejada: 8.5
Venda realizada com sucesso!
```

Questão 2. [2 PONTOS]

Faça um programa que aloca dinamicamente um vetor de strings de tamanho n, onde o valor de n deve ser lido da entrada padrão, seguido das palavras a serem armazenadas no vetor. Em seguida o programa deve ler também uma palavra e verificar quantas vezes essa palavra ocorre no vetor, exibindo o resultado. Ao final da execução o vetor deve ser desalocado.

Exemplos de execução do programa:

```
Tamanho do vetor:
10

Elementos do vetor:
Ana
Maria
Paulo
Rodrigo
Carlos
Ana
Carlos
Vanusa
Ana
Luzia

Digite um nome:
Ana

O nome Ana ocorre 3 vezes.
```

Questão 3. [2,5 PONTOS]

Considere a função que implementa o algoritmo de busca binária apresentada a seguir.

```
int busca_binaria(int v[], int n, int procurado) {
    int esq = 0;
    int dir = n - 1;
    int meio;
    while (esq <= dir) {
        meio = (esq + dir) / 2;
        if (v[meio] == procurado)
            return meio;
        else
            if (v[meio] > procurado)
                dir = meio - 1;
            else
                esq = meio + 1;
    }
    return -1;
}
```

Altere a função para que o valor procurado seja trocado por outro valor, recebido como um parâmetro adicional na função. A declaração da nova função deve ser:

```
int busca_binaria_troca(int v[], int n, int procurado, int novo_valor);
```

Questão 4. [2,5 PONTOS]

A função definida a seguir implementa o algoritmo de busca sequencial, otimizadado para o caso que considera que o vetor sempre já está ordenado com os seus elementos em **ordem decrescente**.

```
bool busca(int vet_ordenado[], int n, int procurado){
    for (int i = 0; i < n; i++) {
        if (procurado == vet_ordenado[i])
            return true;
        if (procurado > vet_ordenado[i])
            return false;
    }
    return false;
}
```

1. Qual é o melhor caso do algoritmo, considerando que o elemento **é encontrado**?
2. Qual é o melhor caso do algoritmo, considerando que o elemento **não é encontrado**?
3. Qual é o pior caso do algoritmo, considerando que o elemento **é encontrado**?
4. Qual é o pior caso do algoritmo, considerando que o elemento **não é encontrado**?