ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
Departamento de Engenharia de Computação e Sistemas Digitais

PCS 3111 - Laboratório de Programação Orientada a Objetos para Engenharia Elétrica

2023

Aula 02 – Ponteiros, Testes e Depuração

Atenção

1. Código inicial para resolução dos exercícios encontra-se **disponível no e-Disciplinas**.
2. Os tipos, os nomes e os parâmetros das funções **devem seguir o especificado** em cada exercício para fins de correção automática.
3. A função `main` **não deve ser submetida**. Caso contrário, a correção automática retornará um *Compilation Error.* **Comente-a antes de submeter.**
4. **Você pode submeter o mesmo arquivo para os dois exercícios.** Só será corrigido o exercício selecionado no Judge (mas, claro, erros de compilação afetam o arquivo como um todo).

Exercício 1
Considere um aplicativo de entregas que deve fazer recomendações de produtos para seus usuários. Cada produto possui três informações: o nome, o número de pedidos e o preço. Essas informações são armazenadas em três vetores diferentes, sempre de mesmo tamanho. Na posição 0 dos vetores estão as informações relativas ao 1º produto, na posição 1 estão as informações relativas ao 2º produto, e assim por diante. Por exemplo, se tivéssemos os seguintes vetores:

```
produtos = {"X-Burger", "X-Salada", "X-Bacon", "X-Tudo"},
numeroDePedidos = {23, 56, 78, 65},
precos = {10, 14.5, 15.90, 20}
```

teríamos quatro produtos:

- “`X-Burger`”, com 23 pedidos e preço 10;
- “`X-Salada`”, com 56 pedidos e preço 14.5;
- “`X-Bacon`”, com 78 pedidos e preço 15.9; e
- “`X-Tudo`”, com 65 pedidos e preço 20.

Deseja-se implementar a seguinte função:

```
string* recomendaProduto(string produtos[], int numeroDePedidos[],
                         double precos[], int quantidade);
```

Essa função deve encontrar o produto com maior número de pedidos. O parâmetro `quantidade` se refere ao tamanho dos três vetores descritos anteriormente (no exemplo anterior, a quantidade era 4).

A função deve retornar um ponteiro com o endereço da posição de memória que armazena o nome do produto recomendado. Caso dois ou mais produtos estejam empatados com o maior

número de pedidos, o produto escolhido deve ser o de **maior preço**. Considere que os produtos não possuem preços iguais.

Usando como exemplo os vetores apresentados anteriormente, a função deve retornar o **ponteiro** com o endereço do produto no índice 2 do vetor **`produtos`** (“`X-Bacon`”), por exemplo `0x00867AB0A0`.

Exercício 2
Parâmetros podem ser usados como entrada e/ou saída de funções. Para criar um parâmetro de saída pode-se usar ponteiros ou passagem por referência. Trabalharemos neste exercício com três tipos de saída: através do retorno da função, através de referências e através de ponteiros. Implemente a função a seguir:

```
double calcularEstatisticas(string categorias[], double precos[],
                            int quantidade, string categoria,
                            double* precoMaximo, double& precoMinimo);
```

Essa função recebe o vetor `categorias`, que possui as diferentes categorias de produtos no aplicativo, e o vetor `precos`, que possui os preços de produtos dessas categorias. Os preços e as categorias estão relacionados de forma análoga ao que foi apresentado no exercício 1. O tamanho de ambos os vetores é definido em `quantidade`. O parâmetro `categoria` indica para qual categoria de produto dentro do vetor `categorias` se deseja realizar os cálculos de preço máximo, mínimo e médio (compare categorias usando ==).

O preço máximo e o mínimo dos produtos da categoria escolhida devem ser retornados pelos parâmetros `precoMaximo` e `precoMinimo`, respectivamente. Atribua o preço máximo dos produtos da categoria escolhida ao valor apontado por `precoMaximo`; atribua o preço mínimo dos produtos da categoria escolhida ao parâmetro `precoMinimo`. O preço médio dos produtos da categoria escolhida é retornado por meio do comando `return`.

Note que `precoMaximo` é passado como um ponteiro e `precoMinimo` é passado como referência.

Assuma que o preço é sempre maior que zero. Para o caso de não haver produtos com a categoria desejada no vetor `categorias`, **todos** os valores retornados devem valer `0`.

Por exemplo, considere os seguintes vetores:

```
categorias = {“Sucos”, “Legumes”, “Sucos”, “Detergentes”},
precos = {3.5, 1.5, 2.9, 5}
```

Considerando que `categoria` seja “Sucos”, teríamos que o valor apontado por `precoMaximo` seria 3.5, `precoMinimo` seria 2.9 e a função retornaria (3.5 + 2.9) / 2 = 3.2.

Cuidado: as variáveis `precoMinimo` e `precoMaximo` não necessariamente são inicializadas antes de serem passadas à função `calcularEstatisticas`. Por exemplo, a função pode ser chamada da seguinte forma:

```
int main() {
    string categorias[] = {“a”, “a”, “b”};
    double precos[] = {1, 2, 3};
    int quantidade = 3;
    string categoria = “a”;
    double precoMinimo; // valor inicial não definido
    double precoMaximo; // valor inicial não definido
    double media = calcularEstatisticas(...); // veja como chamar
    // fim
}
```

No exemplo acima, espera-se que em “fim” tenha-se `precoMinimo=1`, `precoMaximo=2` e `media=1.5`, apesar de os valores iniciais de `precoMinimo` e `precoMaximo` não serem inicializados (e poderem conter “lixo”).

Obs: para realizar uma divisão entre inteiros com resultado real, é necessário fazer um cast para `double` para obter as casas decimais do resultado, conforme o exemplo a seguir:

```
int a = 5, b = 10;
double x = ((double) a) / b;
```

Testes do Judge

Exercício 1

- Teste com apenas um produto no vetor;
- Teste com produto recomendado nas extremidades do vetor (primeira e última posição) e sem produtos com o mesmo número de pedidos;
- Teste com produto recomendado no interior do vetor e sem produtos com o mesmo número de pedidos;
- Teste com produto recomendado nas extremidades do vetor e com produtos com o mesmo número de pedidos;
- Teste com produto recomendado no interior do vetor e com produtos com o mesmo número de pedidos;

Exercício 2

- Teste com vetores vazios;
- Teste com uma categoria desejada;
- Teste com uma categoria indesejada;
- Teste com várias categorias, mas sem nenhuma desejada;
- Teste com várias categorias com apenas uma desejada;

- Teste com várias categorias com duas desejadas;
- Teste com várias categorias com três desejadas;
- Teste com várias categorias com todas desejadas;