Disciplina: Programação Estruturada e Modular

Prof. Carlos Veríssimo
Objetivo: Análise crítica de um código.
Data: 29/11/2024
Autor: Mariana Fernandes Vieira

Melhorias entre o Código Antigo e o Código Novo (feito por mim)

---

1. Incluir Produto

- **Código Antigo:**
  - Aceita ID repetido, o que pode causar confusão.
  - Deixa colocar quantidade ou preço negativo, o que não faz sentido.
  - Se o usuário digitar algo inválido, o programa pode "bugar".
- **Código Novo:**
  - Verifica se o ID já existe antes de adicionar o produto.
  - Só aceita quantidade e preço positivos.
  - Tem validações para garantir que o usuário digite algo válido e não deixe o programa travar.

---

2. Consultar Produto (por ID)

- **Código Antigo:**
  - Mostra que o produto não foi encontrado, mas não valida se o ID é um número válido.
  - Se o ID digitado for inválido, pode acabar gerando resultados inesperados.
- **Código Novo:**
  - Valida se o ID é realmente um número positivo antes de consultar.
  - Dá mensagens mais claras para o usuário se algo der errado.

3. Alterar Produto

- **Código Antigo:**
  - Deixa alterar os valores para negativos (o que não faz sentido).
  - Não valida a entrada, então um erro do usuário pode travar o programa.
  - Não avisa se o ID digitado não existir.
- **Código Novo:**
  - Só aceita valores válidos (quantidade e preço positivos).
  - Dá mensagens mais claras se o ID não for encontrado.
  - Garante que as alterações são feitas de forma segura e válida.

---

4. Excluir Produto

- **Código Antigo:**
  - Se o ID não existir, só diz "Produto não encontrado" e pronto.
  - Não valida se a entrada é realmente um número, podendo dar problemas.
- **Código Novo:**
  - Valida a entrada do ID antes de tentar excluir.
  - Dá uma mensagem mais clara quando o ID não existe.
  - O processo de exclusão é mais organizado e seguro.

---

5. Vender Produto

- **Código Antigo:**
  - Permite vender mais do que tem no estoque, deixando o estoque negativo.
  - Não valida se a quantidade digitada é válida.
  - Pode travar se o usuário errar a entrada.
- **Código Novo:**
  - Não deixa o estoque ficar negativo, verificando se tem quantidade suficiente antes da venda.
  - Valida a entrada do usuário para evitar erros.
  - Dá mensagens mais claras caso algo não esteja certo.

Análise crítica solicitada do código

Modularização:

O código está bem organizado, com funções separadas para cada tarefa, como incluir, consultar, alterar, excluir, vender e imprimir produtos. Isso deixa o código mais fácil de entender e manter. Porém, a parte do main() poderia ser um pouco mais modular. Algumas ações, como a leitura de entradas do usuário, poderiam ser movidas para funções separadas, assim evitaríamos repetir o código várias vezes. Além disso, algumas funções, como incluirProduto(), não retornam nada ou têm um controle de erro mais robusto, o que seria bom de melhorar.

Uso de struct e Ponteiros:

O código usa bem a struct para organizar os dados dos produtos (ID, nome, quantidade e preço), o que ajuda a manter tudo claro e organizado. Quanto aos ponteiros, eles são usados corretamente, especialmente para passar as variáveis entre as funções sem precisar fazer cópias desnecessárias. As funções que alteram o estado do produto (como incluirProduto, alterarProduto e excluirProduto) manipulam os dados diretamente na memória usando ponteiros, o que é uma boa prática. A função consultarProduto também retorna um ponteiro para o produto encontrado, o que evita a cópia da estrutura inteira.

Impressão dos Dados Usando Ponteiro:

A função imprimirProduto() faz certo ao imprimir as informações do produto usando um ponteiro. Isso é legal porque, assim, o código pode acessar os dados diretamente sem precisar fazer cópias da estrutura, economizando memória e deixando o processo mais rápido.