Universidade Federal de Mato Grosso

Bacharelado em Ciência da Computação

Semestre: 2024/1

Programação 2

Lista 8

Exercício 1: Sistema de Gestão de Produtos com Tratamento de Exceções e Namespaces

Descrição:
Implemente um sistema de gestão de produtos que permita o cadastro e a consulta de produtos em um estoque. O sistema deve utilizar namespaces para organizar as classes e funções, além de tratar erros comuns usando exceções adequadas.

Requisitos:

- **Namespaces:** Crie dois namespaces, `Cadastro` e `Consulta`. O namespace `Cadastro` deve conter uma classe `Produto` com métodos para cadastrar produtos, enquanto o namespace `Consulta` deve ter funções para buscar produtos pelo código.
- **Classe Produto:**
  - Deve ter atributos como `codigo`, `nome`, `preco` e `quantidade`.
  - Inclua métodos para cadastrar o produto, obter e atualizar suas informações.
  - Utilize uma exceção `std::invalid_argument` para lançar erros ao tentar cadastrar um produto com código ou nome inválido.
- **Função de Consulta:**
  - Crie uma função `buscarProduto` no namespace `Consulta` que recebe um código de produto e retorna as informações correspondentes.
  - Lance uma exceção `std::out_of_range` se o código do produto não for encontrado.
- **Tratamento de Exceções:**
  - Utilize blocos `try-catch` para capturar as exceções `std::invalid_argument` e `std::out_of_range`.
- **Organização do Projeto:**
  - Organize os arquivos do projeto, separando interface (.h) das implementações (.cpp) e uma main separada, bem como forneça o Makefile para compilar os arquivos.

Objetivo:
Praticar o uso de namespaces, manipulação de exceções e gerenciamento de memória em C++.

Exercício 2: Sistema de Vendas com Exceções e Casting

Descrição:
Desenvolva um sistema de vendas que envolva produtos diferentes, utilizando herança e casting para manipular os produtos e tratando adequadamente os erros.

Requisitos:

- **Classes Base e Derivadas:**
  - Crie uma classe base `Item` com atributos como `codigo`, `preco` e `qtdEstoque`.
  - Crie classes derivadas `Eletronico` e `Vestuario`, que herdam de `Item`.
- **Manipulação de Produtos:**
  - Use `dynamic_cast` para converter ponteiros de `Item` para `Eletronico` ou `Vestuario`.
  - Lance e capture uma exceção `std::bad_cast` se o casting for inválido.
  - Gere uma situação em que a exceção `std::bad_cast` será lançada
- **Sistema de Vendas:**
  - Crie uma função que simule a venda de um produto e lance `std::runtime_error` se a quantidade em estoque for insuficiente.
- **Organização do Projeto:**
  - Organize os arquivos do projeto, separando interface (.h) das implementações (.cpp) e uma main separada, bem como forneça o Makefile para compilar os arquivos.

Objetivo:
Reforçar a aplicação de herança, uso de `dynamic_cast`, e tratamento de múltiplas exceções em C++.

Instruções Adicionais

- Salve cada classe e função em arquivos separados conforme a prática recomendada.
- Utilize comentários para explicar cada parte do código.
- Compile o programa com um `Makefile` adequado.

Exemplo de Código:

```
// Exemplo de codigo em C++ para referencia
namespace Cadastro {
    class Produto {
        // Definicao da classe Produto
    };
}

// Implementacao de funcao de consulta
namespace Consulta {
    Produto buscarProduto(int codigo) {
        // Definicao da funcao buscarProduto
    }
}
```