Exercícios - Programação Orientada a Objetos

Disciplina: Programação Orientada a Objetos
Aluno: Bernardo Henrique Lebron Machado

1.1) Herança múltipla é um mecanismo no qual uma classe pode herdar de duas ou mais classes ao mesmo tempo. Ou seja, uma classe filha recebe atributos e métodos de vários pais simultaneamente. Ela é útil quando um novo tipo precisa combinar comportamentos distintos vindos de classes diferentes.

1.2) O problema do diamante acontece quando uma classe (D) herda de duas classes (B e C) que, por sua vez, herdaram da mesma classe base (A). Se A tem um método com o mesmo nome que B e C sobrescrevem, o compilador não sabe qual versão usar em D, criando ambiguidade e conflito de implementação.

1.3) Com um recurso chamado virtual public, o C++ garante que apenas uma instância de A existe dentro de D, não há ambiguidade de métodos e não há duplicação de atributos.

1.4) É um recurso do Java para não gerar ambiguidades

1.5) Na herança múltipla a classe herda os atributos, métodos e a implementação. Já as múltiplas interfaces não herdam nenhum desses, pois trabalha apenas com o contrato

1.6) Implementação...

2.1) Polimorfismo de sobrecarga é a capacidade de uma classe possuir métodos com o mesmo nome, mas com assinaturas diferentes.

2.2) Em C++, você pode definir valores padrões para parâmetros de uma função ou método. Dessa forma,

se o programador não fornecer aquele argumento, o C++ automaticamente usa o valor padrão.

2.3) Para evitar ambiguidade, manter a linguagem simples e evitar conflitos com herança

2.4) Regra de Posicionamento: devem ser alocados da direita para a esqueda. Regra da Declaração Única: Os valores default devem ser especificados apenas uma vez para uma função. Devem estar na declaração visível do código, ou seja, em algum arquivo .h.

2.5) Implementação...

3.1) Templates é um recurso que permite aos programadores criar funções e classes genéricas, permitindo que o mesmo código funcione com diferentes tipos de dados sem repetição de código.

3.2) Template de função generaliza funções e permite que uma mesma função funcione para vários tipos de dados. Template de classe generaliza estruturas de dados inteiras e é usado para criar classes que funcionam com qualquer tipo.

3.3) As principais vantagens do uso de templates é a reutilização do código e o aumento da flexibilização permitindo que uma única função ou classe trabalhe com qualquer tipo de dado. E suas principais desvantagens é que gera um código mais complicado de entender e aumenta o tempo de compilação

3.4) A instanciação de templates é o processo pelo qual o compilador gera uma função ou classe concreta a partir da definição do template, substituindo os parâmetros de template genéricos pelos tipos de dados reais. Isso acontece exclusivamente em tempo de compilação.

4.1) A principal vantagem é que o vector é um array dinâmico, capaz de crescer e encolher automaticamente conforme você insere ou remove elementos.

4.2) Vector é um array dinâmico, ou seja, uma coleção de elementos armazenados em posições de memória contíguas, cujo tamanho pode crescer ou diminuir automaticamente em tempo de execução. List geralmente se refere a uma lista encadeada, onde os elementos não são armazenados em posições

de memória contíguas. Cada nó contém o valor do elemento e ponteiros para o próximo nó na sequência ou também para o anterior se for uma lista duplamente encadeada. Queue é uma estrutura do tipo fila que segue o princípio FIFO. Funciona como uma fila de pessoas em um banco: a primeira pessoa a entrar na fila é a primeira a ser atendida.

4.3) Deve-se usar stack quando a ordem for LIFO (last in, first out) e usar Queue quando a ordem for FIFO (first in, first out).

4.4) Um iterador é um conceito de programação que atua como um ponteiro inteligente para elementos em um contêiner. Ele fornece uma maneira uniforme de acessar elementos em diferentes tipos de contêineres (vector, list, map) sem expor a estrutura de dados. Ele é importante pois permite percorrer qualquer container da STL da mesma forma, dá acesso controlado aos elementos sem expor como o container é implementado e permite que algoritmos genéricos funcionarem para diferentes containers.

4.5 4.6 e 4.7) Implementação...

5.1) Uma exceção é um mecanismo usado para sinalizar que ocorreu um erro ou situação anormal durante a execução do programa. Deve ser usada quando acontece alguma entrada inválida, quando se quer separar lógica normal da lógica de erro e quando o erro precisa ser tratado de forma segura e organizada.

5.2) Try é utilizado para definir blocos de código em que excessões podem ocorrer. Throw é a instrução que indica que houve um erro. Catch é o responsável por imprimir o erro e tentar refazer a operação.

5.3) Em exceções verificadas o compilador orbiga a tratar try, catch ou a declarar throws e podem ser previstas em tempo de compilação. Exceções não verificadas não precisam ser tratadas obrigatoriamente e geralmente ocorrem por erro de lógica.

5.4) Finally é um bloco opcional que tem o objetivo de executar um código de limpeza ou liberação de recursos, independentemente de uma exceção ter ocorrido ou não. É sempre executado.

5.5 5.6) Implementação...

6) Implementação...