

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Engenharia de Software – Manhã - Campus Lourdes

Laboratório de Algoritmos e Estruturas de Dados 1 – 1/2025

AULA 8 – PROGRAMAÇÃO ORIENTADA A OBJETOS

Observações:

- Implemente os programas utilizando a linguagem C++.
- Na resolução dos exercícios só podem ser utilizados comandos vistos nas aulas.

Exercícios:

Desenvolva algoritmos e programas para os seguintes desafios e implemente-os em C++:

1. Classe Simples

Crie uma classe chamada Pessoa com dois atributos: nome e idade. Escreva um programa que instancie um objeto dessa classe, atribua valores e imprima-os no terminal.

2. Construtores e Métodos

Altere a classe Pessoa do exercício 1 para incluir um **construtor** que receba nome e idade como parâmetros. Adicione também um método chamado `exibirDados()` que mostre as informações da pessoa.

3. Encapsulamento

Modifique a classe Pessoa do exercício 2 para que os atributos sejam privados. Crie métodos `get` e `set` para acessar e modificar os dados da pessoa com segurança.

4. Classe Produto

Crie uma classe Produto com atributos privados: nome, preço e quantidade. Implemente os métodos: `cadastrarProduto`, `mudarPrecoProduto`, `mudarQuantidadeProduto` e `calcularValorTotalEmEstoque`.

5. Classe Conta Bancária

Crie a classe `ContaBancaria` com atributos: número da conta, nome do titular e saldo. Implemente métodos para depositar, sacar (com verificação de saldo) e `mostrarSaldo`.

Crie um menu para perguntar ao usuário se ele deseja realizar depósito, saque, mostrar saldo ou finalizar o programa.

No método `sacar`, controle para não deixar o saldo ficar negativo.

6. Múltiplos Objetos

Utilizando a classe `Produto` do exercício 4, crie um vetor de objetos para armazenar uma lista de 5 produtos. Implemente um menu simples para cadastrar os produtos e exibir todos.

7. Classe Data

Implemente uma classe `Data` com atributos dia, mes e ano. Crie métodos para validar a data e imprimir no formato `dd/mm/aaaa`.

8. Composição de Classes

Crie uma classe `Endereco` (rua, cidade, CEP). Em seguida, atualize a classe `Pessoa` para conter um atributo do tipo `Endereco`. Implemente métodos para exibir os dados completos da pessoa com o endereço.

9. Herança Simples

Crie uma classe base `Funcionario` com atributos nome e salário. Depois, crie uma classe derivada `Gerente` que herda de `Funcionario` e adiciona o atributo departamento.

10. Herança com Construtores

Expanda o exercício anterior adicionando construtores nas duas classes (`Funcionario` e `Gerente`) e métodos para exibir os dados de cada um.

11. Polimorfismo com Métodos

Crie a classe `Funcionario` com um método `calcularBonus()` que retorna 10% do salário. Na classe `Gerente`, sobrescreva esse método para retornar 20% do salário.

12. Classe Abstrata

Crie uma classe abstrata `FormaGeometrica`. Crie duas classes derivadas: `Retangulo` e `Circulo`. Implemente o cálculo da área em cada classe.

13. Vetores e Polimorfismo

Utilize o exercício anterior para criar um vetor para `FormaGeometrica`, armazenando objetos `Retangulo` e `Circulo`. Percorra o vetor e chame `calcularArea()` para cada forma.

14. Sistema de Cadastro com Herança

Implemente um sistema simples de cadastro com as classes `Pessoa`, `Aluno` (com matrícula e curso) e `Professor` (com disciplina e carga horária). Use herança e polimorfismo para imprimir os dados.

15. Projeto Final – Sistema Bancário

Crie um sistema com as seguintes classes:

- `Cliente`: nome, CPF.
- `Conta` (classe abstrata): número, saldo, métodos `depositar()`, `sacar()`, `exibirSaldo()`.
- `ContaCorrente` e `ContaPoupanca`: derivadas de `Conta`, com regras específicas (ex: taxa de saque para corrente, rendimento para poupança).

Implemente um menu para criar contas, realizar transações e listar os dados das contas cadastradas.