Universidade Federal de Mato Grosso

Bacharelado em Ciência da Computação

Semestre: 2024/2

Programação 3

Lista 2

Exercício 1: Construção de Objetos e Passagem de Parâmetros

Crie uma aplicação que implemente as seguintes funcionalidades:

1. **Classe `Car`**:
   - Defina os atributos `marca` (String), `modelo` (String) e `ano` (int).
   - Implemente três construtores:
     - Um que inicialize todos os atributos.
     - Um que inicialize `marca` e `modelo` com valores fornecidos, definindo `ano` como 2000.
     - Um que inicialize todos os atributos com valores padrão.
   - Crie um método `exibirDetalhes()` que exibe as informações do carro.
2. **Classe `Garage`**:
   - Crie uma classe que gerencia um conjunto de objetos `Car` usando um array.
   - Adicione os atributos `countVW` e `countFord` que incremente toda vez que veículos desta marca forem adicionados à garagem.
   - Adicione métodos para:
     - Adicionar um carro à garagem.
     - Exibir os detalhes de todos os carros armazenados.
     - Exibir a contagem geral e das marcas VW e Ford.
3. **Construtor de Cópia**:
   - Na classe `Car`, adicione um construtor de cópia que permita criar um novo objeto a partir de um existente.
4. **Teste no `main()`**:
   - Crie alguns objetos de `Car` usando os diferentes construtores.
   - Adicione esses carros à garagem.
   - Copie um dos carros existentes usando o construtor de cópia.
   - Exiba a contagem e as informações de todos os carros da garagem.

Exercício 2: Varargs e Cálculos Estatísticos

Implemente uma classe que modele um experimento científico com um número variável de resultados.

- Crie a classe `Experimento`:
  - Atributos:

        * `nome` (String): Nome do experimento.
        * `resultados` (double[]): Resultados do experimento.
    - Métodos:
        * Construtor que aceita um nome e um número variável de resultados (usando `varargs`).
        * `calcularMedia()`: Retorna a média dos resultados.
        * `calcularDesvioPadrao()`: Retorna o desvio padrão dos resultados.

- Crie uma classe de teste:
    - Instancie pelo menos três experimentos com diferentes números de resultados.
    - Para cada experimento, exiba o nome, a média e o desvio padrão.

Exercício 3: Classes Internas e Funcionalidades Aninhadas

Modele uma organização com departamentos e funcionários, usando classes internas para representar funcionários.

- Classe `Departamento`:
    - Atributos:
        * `nome` (String): Nome do departamento.
        * Classe interna `Funcionario`:
            · `nome` (String): Nome do funcionário.
            · `cargo` (String): Cargo do funcionário.
            · `salario` (double): Salário do funcionário.
            · Método `exibirDetalhes()`: Exibe os detalhes do funcionário.
    - Métodos:
        * `adicionarFuncionario(String nome, String cargo, double salario)`: Adiciona um funcionário ao departamento.
        * `listarFuncionarios()`: Lista todos os funcionários do departamento.

- Classe `Organizacao`:
    - Atributos:
        * Lista de departamentos.
    - Métodos:
        * `adicionarDepartamento(String nome)`: Adiciona um departamento à organização.
        * `listarDepartamentos()`: Lista os departamentos e seus funcionários.

- Classe de teste:
    - Crie uma organização com pelo menos dois departamentos (*e.g.*, TI e RH).
    - Adicione funcionários a cada departamento.
    - Liste os departamentos e seus respectivos funcionários.

Exemplo no `main()`:

```
public class Main {
    public static void main(String[] args) {
     Organizacao org = new Organizacao();

     Departamento ti = org.adicionarDepartamento("TI");
     ti.adicionarFuncionario("Joao", "Desenvolvedor", 5000.0);
     ti.adicionarFuncionario("Ana", "Analista", 6000.0);

     Departamento rh = org.adicionarDepartamento("Recursos Humanos");
     rh.adicionarFuncionario("Carlos", "Gestor", 4500.0);

     org.listarDepartamentos();
    }
}
```