Atividade: Sistema de Gestão de Funcionários em uma Empresa

Objetivo: Criar um sistema simples de gestão de funcionários usando os conceitos de herança, construtores sobrecarregados, listas e agregação.

Contexto:

Você foi contratado para desenvolver um sistema de gerenciamento de funcionários para uma empresa. O sistema deverá ser capaz de gerenciar diferentes tipos de funcionários (efetivos e terceirizados), seus dados e as equipes a que pertencem.

Requisitos:

1. **Classes Base:**
    - **Pessoa:** Classe que representa uma pessoa genérica com atributos e métodos comuns.
        - Atributos: `nome`, `idade`
        - Métodos: `exibirDados()`
        - Construtores: Um construtor que recebe o nome e idade como parâmetros e outro construtor que inicializa com valores padrão.
2. **Herança:**
    - **Funcionario (herda de Pessoa):** Classe que representa um funcionário da empresa.
        - Atributos: `salario`, `dataContratacao`
        - Métodos: `calcularBonus()`, que retorna o valor do bônus do funcionário (fixo de 10% do salário)
        - Construtores: Um construtor que recebe os parâmetros da pessoa, salário e data de contratação, e outro que inicializa apenas com o nome e salário, definindo a data de contratação como a data atual.
    - **Terceirizado (herda de Funcionario):** Classe que representa um funcionário terceirizado.
        - Atributos: `empresaContratada`
        - Métodos: O método `calcularBonus()` deve ser sobrescrito para retornar um valor fixo de bônus de 5%.
        - Construtores: Um construtor que recebe os parâmetros da pessoa, salário, data de contratação e empresa contratada, e outro que inicializa apenas com o nome, salário e empresa contratada.
3. **Agregação e Listas:**
    - **Equipe:** Classe que representa uma equipe na empresa.
        - Atributos: `nomeEquipe`, `lider` (que é um objeto da classe Funcionario), e `membros` (uma lista de funcionários que pertencem à equipe).
        - Métodos:

        - `adicionarMembro(Funcionario membro)`: Adiciona um funcionário à equipe.
        - `removerMembro(Funcionario membro)`: Remove um funcionário da equipe.
        - `exibirMembros()`: Exibe os dados de todos os membros da equipe.
    - Construtores: Um construtor que recebe o nome da equipe e o líder da equipe como parâmetros.

Tarefa:

Implemente as classes conforme os requisitos acima. Ao final, crie um programa principal (método `main`) que execute as seguintes ações:

1. Crie dois objetos da classe `Funcionario` (um com todos os parâmetros e outro usando o construtor com valores padrão).
2. Crie dois objetos da classe `Terceirizado` (um com todos os parâmetros e outro usando o construtor com valores padrão).
3. Crie um objeto da classe `Equipe` e defina um funcionário como líder.
4. Adicione todos os funcionários e terceirizados à equipe.
5. Exiba os membros da equipe, chamando o método `exibirDados()` de cada membro.
6. Calcule e exiba o bônus de cada membro da equipe.

Dicas para implementação:

- Use sobrecarga de construtores para facilitar a criação de objetos com diferentes dados de entrada.
- Lembre-se de utilizar herança para reutilizar código entre as classes.
- A agregação será representada pela relação entre a equipe e seus membros, que são objetos da classe `Funcionario` ou suas subclasses.