Questão 2) Reflita e Responda com suas palavras as seguintes perguntas:

1. O que é herança em Programação Orientada a Objetos (POO) e qual seu principal objetivo?

Herança em POO é quando uma classe filha pega atributos e métodos de uma classe mãe" O principal objetivo é reutilizar código, torná-lo mais enxuto. Afinal, utilizando esse método, não há a necessidade de escrever a mesma coisa várias vezes.

2. Qual é a diferença entre uma superclasse e uma subclasse? Dê um exemplo de cada.

Durante o próprio código da questão 1, houve um exemplo de superclasse e subclasse. O método exibirDados() esteve presente em ambas as subclasses e por isso teve FuncionarioTi dado como superclasse. Isso é, o mesmo método, ainda que tenha sofrido override, estava presente na classe-mãe (FuncionarioTI) e foi herdado às filhas.

3. Dada a seguinte hierarquia UML:
   Pessoa → Estudante,
   o que significa afirmar que "todo Estudante é uma Pessoa, mas nem toda Pessoa é um Estudante"?

Pessoa diferencia-se em várias funções. Pode dar caminho tanto a formação dum Estudante, quanto do Professor. Isso é, tanto Professor quanto Estudante são partes do conjunto Pessoa, mas o inverso não é verdadeiro.

4. Quais são as vantagens do uso da herança no desenvolvimento de software orientado a objetos?

A vantagem está na organização e leitura do código, encurtamento e segurança dos dados.

5. Qual o símbolo e a direção da seta usada para representar herança em diagramas de classes UML?

Seta vazada de ponta triangular ligando das filhas à mãe.

6. Para que serve a palavra-chave super em Java? Cite dois contextos diferentes em que ela pode ser usada.

Refere-se a algum parâmetro (1) ou atributo (2) da superclasse. Pode ser utilizada tanto para complementar um código na subclasse quanto para reaproveitar informações anteriormente coletadas na superclasse.

7. Um sistema define as classes Professor, ProfHorista e ProfDE. Por que é mais vantajoso centralizar atributos comuns na classe Professor ao invés de declará-los separadamente nas subclasses?

Além da facilidade sintática, Horista e DE são ramificações de professor. Os objetos de ambas as classes com certeza têm elementos comuns da classe professor: nome, horário, disciplinas, turmas, dentre outros.

8. Suponha que você esteja modelando um sistema de transporte. Como você organizaria uma hierarquia de classes para representar Transporte, TransporteTerrestre, TransporteAereo, Carro, Avião e Helicóptero? Qual o papel da herança nessa modelagem?

Transporte
- TransporteAereo
    - Avião
    - Helicóptero
- TransporteTerrestre
    - Carro

Helicóptero e Avião utilizariam da mesma informação contida em TransporteAereo, a qual difere-se em partes da utilizada em TransporteTerrestre. Entretanto, tanto TransporteAereo quanto TransporteTerrestre são transportes e, portanto, pertencem à classe Transporte.

9. O que acontece se, em uma subclasse, você não chamar explicitamente o construtor da superclasse usando super()? Em que situação isso pode gerar erro?

Caso não se chame super() explicitamente no construtor da subclasse, o Java tenta chamar o construtor padrão (sem argumentos) da superclasse automaticamente. Isso pode gerar erros se a superclasse não tiver um construtor padrão (sem argumentos). Por exemplo, se a superclasse só tiver construtores que recebem parâmetros, tipo FuncionarioTI(nome, matricula, funcao), a subclasse é obrigada a chamar um desses construtores da superclasse com super(...), senão o Java não sabe como inicializar a parte da superclasse do objeto e, portanto, gera erro de compilação.