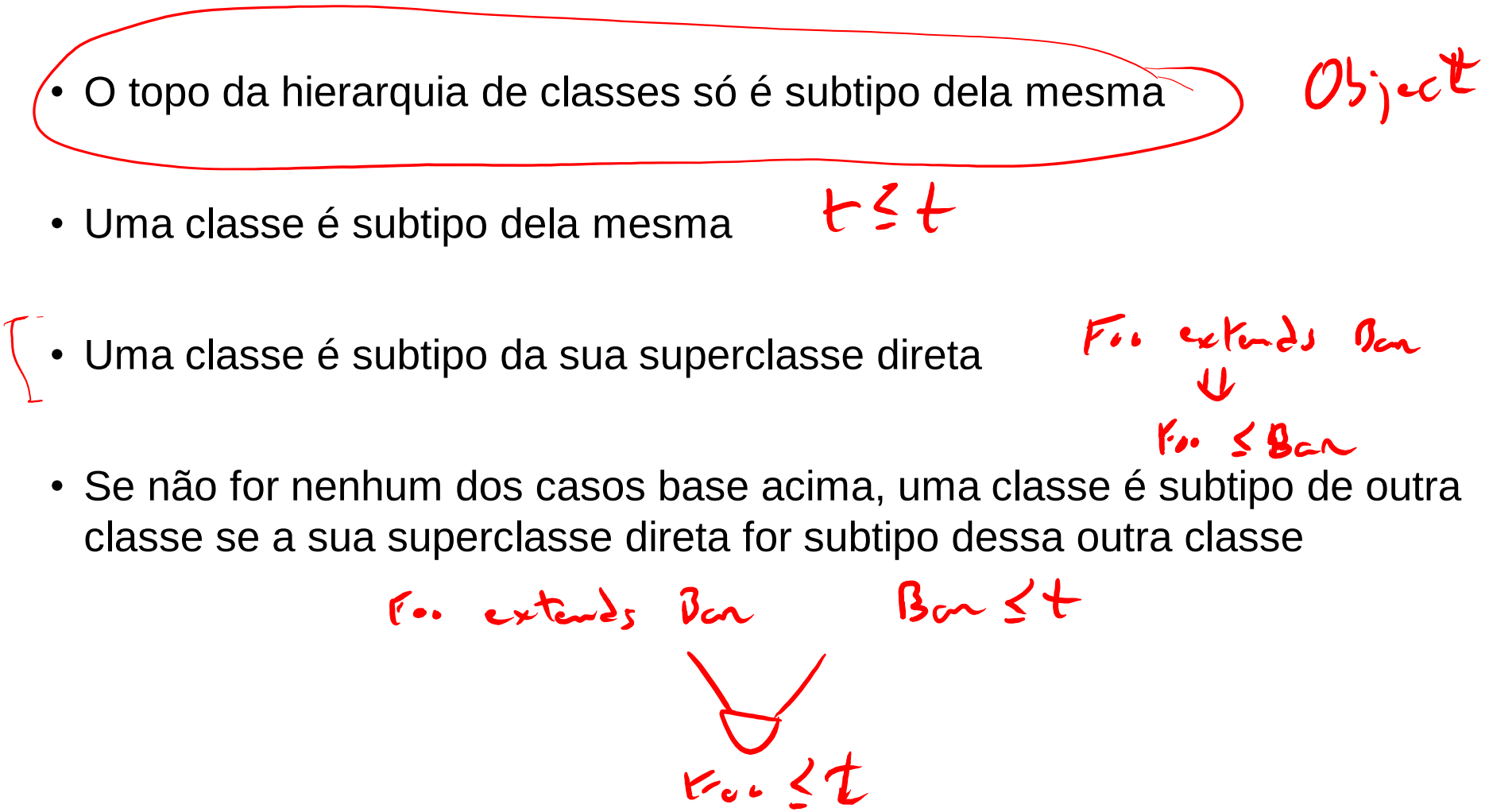
Tipagem em MiniJava – classes e métodos

- Em MiniJava, quando o programador define uma classe ele está adicionando um novo tipo à linguagem
- Uma classe é só uma versão mais complicada dos tipos de procedimento que vimos para TINY
- Ele tem quatro partes: o nome da classe, o nome da superclasse, os seus campos e os tipos associados, e os seus métodos e tipos associados
- O tipo de cada método é como o tipo dos procedimentos de TINY

Subtipagem em MiniJava

- Determinar se uma classe é subtipo de outra é fácil com um algoritmo recursivo
- O topo da hierarquia de classes só é subtipo dela mesma
- Uma classe é subtipo dela mesma
- Uma classe é subtipo da sua superclasse direta
- Se não for nenhum dos casos base acima, uma classe é subtipo de outra classe se a sua superclasse direta for subtipo dessa outra classe



Redefinição de métodos

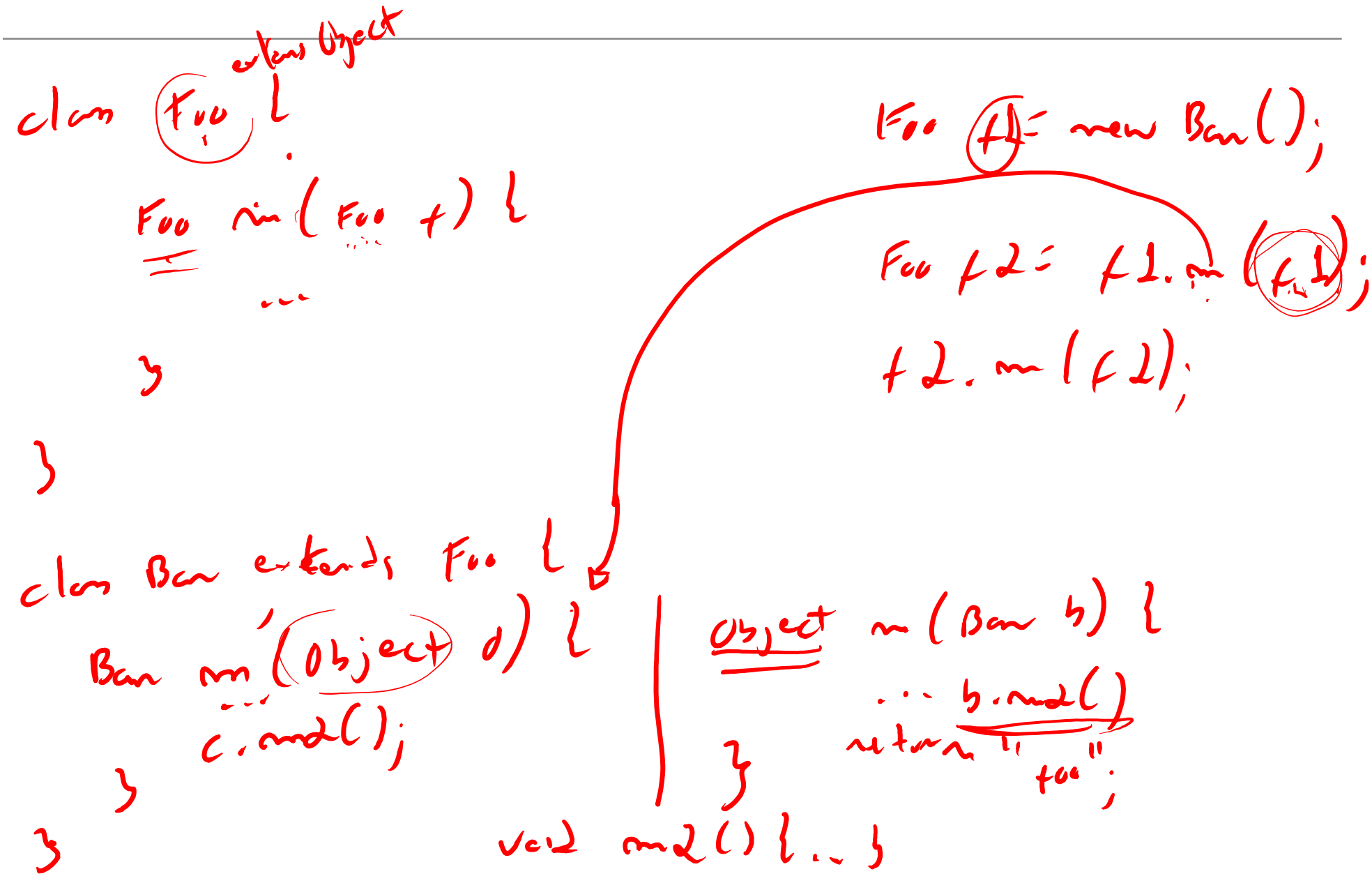
- Quando redefinimos um método em uma subclasse, poderíamos permitir que os tipos dos parâmetros e o tipo de retorno mudem

- O tipo dos parâmetros na subclasse poderiam ser *supertipos* dos tipos na superclasse, e o tipo de retorno poderia ser um *subtipo*

- Por isso dizemos que métodos são *contravariantes* no tipo dos parâmetros e *covariantes* no tipo de retorno

- É fácil ver que parâmetros covariantes seriam inconsistentes, assim como um tipo de retorno contravariante

Contravariância e covariância



Contextos de tipagem em MiniJava

- Ao contrário de TINY, a verificação de tipos para expressões e comandos MiniJava precisa de um único contexto de tipagem
- Chamadas de métodos obtêm o tipo do método a partir da classe do objeto alvo
- A classe associada a `this` pode ficar no mesmo contexto das outras variáveis e campos
- A verificação precisa de duas passadas: a primeira cria as classes, sem se importar com a consistência dos tipos nos corpos dos métodos, e a segunda verifica os corpos dos métodos