UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Campus CAMPO MOURÃO

Resenha Capítulo 2 - APOO

Herança

Borges e Clinio

Estudante: Reginaldo Gregório de Souza Neto RA: 2252813

A herança é um conceito que se aplica única e exclusivamente às classes, permitindo com que se construa e estenda continuamente outras classes desenvolvidas anteriormente. O objetivo da herança é tornar as classes cada vez mais simples, fazendo com cada uma execute apenas o seu papel principal ao qual lhe foi atribuído inicialmente. Cada vez que se deriva uma nova classe começando de uma já existente, pode-se herdar algumas ou todas as características da classe pai, adicionando novas quando for necessário. Sendo assim, é possível herdar todos os comportamentos das classes bases.

Com a herança é possível criarmos uma classe PESSOA por exemplo, com os atributos de NOME e IDADE, e logo em seguida criar uma classe "filha" de pessoa chamada BRASILEIRO, com o atributo CPF. Nesse caso, ao acessar os atributos de um objeto BRASILEIRO, é possível observar que NOME, IDADE e CPF podem ser manipulados da mesma maneira. Sem a necessidade de uma instrução formal, dizendo que os atributos NOME E IDADE foram herdados.

Nem tudo é herdado quando se declara uma classe derivada. Alguns casos são inconsistentes com herança por definição:

- Construtores
- Destrutores
- Operadores new
- Operadores de atribuição (=)
- Relacionamentos friend
- Atributos privados

Classes derivadas invocam o construtor da classe base automaticamente, assim que são instanciadas.

Além dos especificadores public e private, com a herança é possível adicionarmos mais um chamado PROTECTED, que assim como o private "oculta" o atributo para classes e escopos externos, porém é possível acessá-lo de classes filhas.

Os construtores são invocados quando se deseja instanciar uma classe, e no caso da herança eles precisam ser chamados de acordo com a herança entre as classes, ou seja, primeiro se chama o construtor da classe "pai" e depois da classe "filha" e assim sucessivamente em caso

de heranças múltiplas. Isso se deve ao fato de que é possível manipular atributos de classes pais nas classes filhas, portanto se faz necessário a construção prévia desses atributos em ordem “cronológica”. Já os destrutores por sua vez, possuem o papel inverso dos construtores, assim como sua sequência de invocação também é invertida. Quando se destrói alguma classe, chama-se primeiro os destrutores das classes filhas e em seguida é chamado o destrutor da classe em questão.

É possível declarar classes com os especificadores public e private, sendo que no public os atributos continuam com suas mesmas características na classe derivada. E no private todos os atributos se tornam private na classe derivada.