ESQUEMA AULA PRÁTICA 10

❑ Herança, Polimorfismo e Ligação em Java

Podem definir-se novas classes a partir de classes já existentes. A nova classe, classe derivada ou subclasse, vai herdar as características e funcionalidades da classe base ou superclasse, podendo redefinir funções e/ou adicionar novos membros. As restrições de acesso são estendidas à subclasse e seus clientes. Enquanto os membros da classe derivada não têm acesso aos membros privados da classe base, os membros públicos da classe base são automaticamente membros da classe derivada.

- Suponhamos a classe **C1**, definida abaixo, que contém dois métodos **m1 e m2**:

```
public class C1 {
   public void m1() {
     System.out.println ("Método 1 classe C1");
   }
   public void m2() {
     System.out.println ("Método 2 classe C1");
   }
}
```

- Suponha também uma subclasse de C1, a classe **C2** em que o método **m2** é redefinido:

```
public class C2 extends C1{
    public void m2() {
     System.out.println ("Método 2 classe C2 ");
    }
}
```

Os membros m1 e m2 são métodos polimórficos que podem ser aplicados a vários tipos de objectos: m1 ilustra o polimorfismo de inclusão, pois pode ser invocado em objectos do tipo C1 e qualquer subtipo deste. O tipo de objecto em que o método m2 for invocado determinará a implementação do método que vai ser executada (a este tipo de polimorfismo chama-se polimorfismo de sobreposição ou "overrinding").

1. Qual será o output da execução do seguinte segmento de código?

```
public static void main (String str[]) {
    C1 o1= new C1();
    C2 o2= new C2();
    o2.m1();
    o2.m2();
    o1.m2();
}
```

- Implemente o exemplo para verificar a sua resposta.

2. Suponha agora que definíamos um método, f, que tem como parâmetro um objecto do tipo C1,

```
void f (C1 x1){
    x1.m2();
  }
```

e que invocamos a função, respectivamente com um objecto da classe C1 e classe C2:

```
f (o1);
f (o2);
```

- Qual acha que seria o output dessa execução?

Faça dois testes:

a) Primeiro, declare a função dentro da classe de teste e invoque-a no método main.

b) Segundo, defina uma classe C com o método f, e seguidamente na classe de teste declare um objecto da classe C, e invoque o método f sobre esse objecto.

- O que pode concluir quanto ao mecanismo de ligação da linguagem, é estático ou dinâmico?

3. Desenvolva um sistema de Gestão de Contas Bancárias[1].
O mesmo deverá ser capaz de realizar diversas operações bancárias sobre contas, tendo em atenção que existem 3 tipos de conta diferentes: normal, vencimento e a prazo.
Todas as contas têm um número, um ou mais titulares, uma data de criação e um saldo.

As contas normais não permitem que o saldo seja inferior a zero.
As contas vencimento permitem que o saldo seja negativo (até determinado limite que pode ser alterado) e mantêm o número de identificação bancária (NIB) da entidade empregadora.
A conta a prazo rende juros, dependendo do tempo que cada depósito permaneça na conta.
Sobre todas as contas pretende-se ter disponíveis as seguintes operações:

a. Criação de conta
b. Levantamento de uma quantia
c. Depósito de uma quantia
d. Consulta do saldo actual
e. Consulta da data, montante e descrição dos últimos 5 movimentos (depósitos e levantamentos)

Identifique os objectos relevantes para o seu sistema e implemente as classes necessárias servindo-se do mecanismo de herança sempre que apropriado.

4. Teste as classes anteriores.

[1] Exercício adaptado de "Programação Orientada aos Objectos em Java 2", *F. Mário Martins* , FCA, 2000.