

LISTA 3: HERANÇA

1. Considere as seguintes classes:

```
class Ponto {  
    double x,y;  
  
    public void limpa() {  
        this.x = 0;  
        this.y = 0;  
    }  
}
```

```
class Pixel extends Ponto {  
    Color cor;  
}
```

Qual a diferença entre elas, em termos do que elas fazem?

2. Considere as seguintes classes:

```
public class A {  
    int x;  
    public A(int x) { this.x = x;}  
}
```

```
public class B extends A {  
    float y;  
    public B(float y) { this.y = y;}  
}
```

Ela irá compilar? Por que? O que você pode fazer para consertar isso?

3. Figuras geométricas são um bom exemplo para se construir um hierarquia de classes. Considere as figuras quadrado, retângulo, triângulo (retângulo, acutângulo e obtusângulo) e losango. Construa duas diferentes hierarquias para estas classes, uma com dois níveis e outra com três níveis, colocando os atributos que devem ser considerados comuns em cada nível da hierarquia. Compare estas duas hierarquias, discutindo suas vantagens e desvantagens (Goldman & Silva, Ex. 2).
4. Desenvolva um conjunto de classes para controlar o saldo, depósitos e retiradas de contas bancárias bem como os dados do titular. Escreva inicialmente um diagrama modelando tanto contas corrente quanto contas poupança e aplicações em fundo. Em seguida, implemente estas classes em Java (Goldman & Silva, Ex. 3).
5. Muitas vezes, em cálculos com datas, precisamos levar em conta não o número de dias corridos, mas sim o número de dias úteis. Assim:
- Construa uma classe, chamada Dia, com os atributos dia, mês e ano, e os seguintes métodos:
 - int diasEntre(Dia d): recebe um objeto Dia e calcula o número de dias corridos entre a data em d e a data dos atributos do objeto de onde este método foi chamado. Ex:
Dia d1 = new Dia(10,5,2010);
Dia d2 = new Dia(15,6,2011);
d1.diasEntre(d2);
 - Dia soma(int nDias): recebe um determinado número de dias a ser somado à data corrente, retornando uma nova data, correspondendo à data corrente + nDias. Se nDias for negativos, subtrai, em vez de somar
 - Construa uma classe, chamada DiaUtil, com os atributos dia, mês e ano, e os seguintes métodos:

- `int diasEntre(Dia d)`: recebe um objeto `Dia` e calcula o número de dias úteis entre a data em `d` e a data dos atributos do objeto de onde este método foi chamado. Para simplificar, leve em conta apenas sábados e domingos. Ex:
`DiaUtil d1 = new DiaUtil(10,5,2010);`
`Dia d2 = new Dia(15,6,2011);`
`d1.diasEntre(d2);`
- `DiaUtil soma(int nDias)`: recebe um determinado número de dias úteis a ser somado à data corrente, retornando uma nova data, correspondendo à data corrente + `nDias`. Se `nDias` for negativos, subtrai, em vez de somar
- Defina uma hierarquia entre essas classes, determinando quem é subclasse de quem, e isolando o que elas têm em comum (podendo haver sobrecarga e sobrescrita de métodos, inclusive)
- Escreva um programa que crie dois objetos para data (útil ou corrida, decida), dizendo o número de dias úteis e corridos entre elas. O que precisa para fazermos isso com apenas esses objetos?