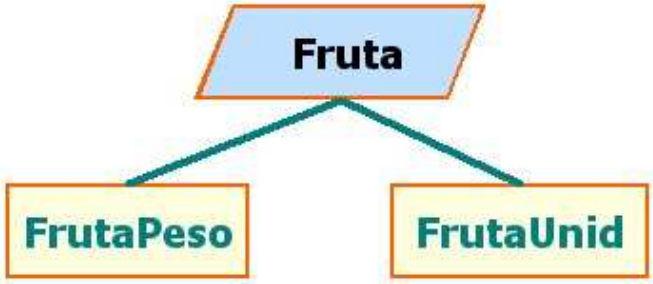
# HIERARQUIA E POLIMORFISMO (2)




```
public abstract class Fruta {
  private double preco;
  private String nome;
  //
  public Fruta(String nm, double p) {
    nome = nm; preco = p);
  }
  public Fruta(Fruta f) {
    nome = f. getNome(); preco = f.getPreco();
  }
  public String getNome() { return nome; }
  public double getPreco() { return preco; }
  //
  public abstract String toString();
  public abstract Fruta clone();
  public abstract double aPagar();
}
```

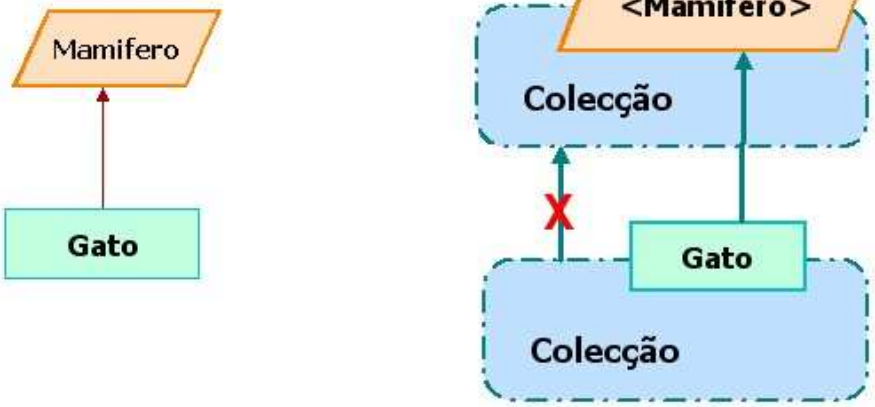
```
public class FrutaPeso extends Fruta {
   private double peso;
  //
  public FrutaPeso(String nm,
                        double pr, double ps) {
    super(nm, pr); peso = ps;
  }
  public FrutaPeso(FrutaPeso f) {
    super(f. getNome(), f.getPreco());
    peso = f. getPeso();
  }
  //
  public double getPeso() { return peso; }
  public double aPagar() {
     return peso*super.getPreco();
  }
  public String toString() { ..... };
  public FrutaPeso clone() {
      return new FrutaPeso(this);
  }
}
```

```
public class FrutaUnid extends Fruta {
   private int quant;
  //
  public FrutaUnid(String nm,
                        double pr, int qtd) {
    super(nm, pr); quant = qtd;
  }
  public FrutaUnid(FrutaUnid f) {
    super(f. getNome(), f.getPreco());
    quant = f. getQuant();
  }
  //
  public int getQuant() { return quant; }
  public double aPagar() {
     return quant*super.getPreco();
  }
  public String toString() { ..... };
  public FrutaUnid clone() {
      return new FrutaUnid(this);
  }
}
```

```
public class Cabaz {
  private ArrayList<Fruta> cabaz = new ArrayList<Fruta>();
....
  // Juntar uma Fruta ao cabaz
  public void junta(Fruta f) { cabaz.add(f.clone()); }
  // Valor total a pagar pelo cabaz
  public double aPagar() {
    double total = 0.0;
    for(Fruta f : cabaz) total += f.aPagar();
   return total;
  }
  // Total de frutos por peso
  public int numFrutosPorPeso() {
    int total = 0;
    for(Fruta f : cabaz)
        if(f instanceof FrutaPeso) total ++;
   return total;
  }
  // Conjunto dos nomes dos frutos do cabaz
  public TreeSet<String> nomesFrutos() {
    TreeSet<String> nomes = new TreeSet<String>();
    for(Fruta f : cabaz) nomes.add(f.getNome());
    return nomes;
  }
  // Total de frutos à unidade comprados
  public int totalFrutosUnidade() {
    int total = 0;
    for(Fruta f : cabaz)
        if(f instanceof FrutaUnid)
                total += ((FrutaUnid) f).getQuant();
   return total;
  }
  // Junta frutos ao cabaz
  public void juntaAoCabaz(ArrayList<Fruta> cab) {
    for(Fruta f : cab) cabaz.add(f.clone());
  }
```

**Questão a ver: Podemos atribuir a um ArrayList<Fruta> um ArrayList<FrutaPeso> ??**

# COLECÇÕES DE JAVA NÃO SÃO CO-VARIANTES




```
ArrayList<Mamifero> mamif = new ArrayList<Mamifero>();
ArrayList<Gato> gatos = new ArrayList<Gato>();
gatos. add( new Gato("TIKO", "X", 3.5) ); .....
mamif = gatos;  // ERRO DE COMPILAÇÃO !
```

**SOLUÇÃO:**

**UTILIZAÇÃO DE WILDCARDS !!**

**Em vez de escrevermos:**

```
public void juntaMamif(Collection<Mamifero> cm) {
  ...
}
```

**usamos o wildcard ? extends Mamifero que generaliza o tipo**

```
Collection<? extends Mamifero> compatível com,
     Collection<Gato>  ou
     Collection<Cao>   ou
     Collection<Coelho>

public void juntaMamif(Collection<? extends Mamifero> cm) {
  ...
}
```