Compartilhar o seu link com: luciorocha @ professores.utfpr.edu.br

```
public class Principal {

    public abstract class Animal {
        public abstract void caminhar();
    }
    public class Sapo extends Animal {
        public void caminhar() {   System.out.println("PULA");  }
    }
    public class Leao extends Animal {
        public void caminhar() {  System.out.println("ANDAR"); }
    }

   public Principal(){
         Animal  animal;
         animal = new Sapo();       //superclasse ←subclasse
         animal.caminhar();

         animal = new Leao();
         animal.caminhar();


         Leao leao = (Leao) animal;  //subclasse ← superclasse (coercao)
   }

   public static void main( String [ ] args ){
        new Principal();
   }

}
```

Exemplo: Software de cadastro de bicicletas. O sistema cadastra bicicletas do tipo montain bike, Speed, ergonômica.

Nomes (classes): Bicicleta, MontainBike, Speed, Ergonomica
Atributos (variáveis de instância):
Verbos (métodos): cadastrar

```
public class Principal {

    public abstract class Bicicleta {
        protected float aro;
        public Bicicleta( float aro){
            this.aro = aro;
        }
        public abstract float getAro();
        public abstract float cadastrar();
    }
    public class MontainBike extends Bicicleta {
        public MontainBike(float aro){
            super( aro );
        }
        public float getAro(){ return this.aro };
        public float cadastrar(){ return getAro(); }
    }
    public class Speed extends Bicicleta {
        public Speed(float aro){
            super( aro );
        }
        public float getAro(){ return this.aro };
        public float cadastrar(){ return getAro(); }
    }
   public class Ergonomica extends Bicicleta {
        public Ergonomica(float aro){
            super( aro );
        }
        public float getAro(){ return this.aro };
        public float cadastrar(){ return getAro(); }
    }

    public void iniciar(){
            ArrayList<Bicicleta>  lista = new ArrayList<>();

            Bicicleta bicicleta = new MontainBike( 15.0f );
            lista.add( bicicleta );

            bicicleta = new Speed( 20.0f );  //polimorfismo
            lista.add( bicicleta );

            bicicleta = new Ergonomica( 25.0f );   //polimorfismo
            lista.add( bicicleta );

            for( Bicicleta  b : lista  )
                System.out.println( b.cadastrar() );
    }
```

```
    public static void main( String [ ] args ){
            Principal principal = new Principal();
            principal.iniciar();
     }

}
```

//Exemplo 2:

```
Animal animal = new Sapo();  //OK. Sapo eh um Animal
Sapo sapo = new Animal();     //Não OK. Animal não eh um Sapo.

MembroUniversitário membro = new Professor();  //OK. Professor eh um
                                                //       MembroUniversitario
Professor professor = new MembroUniversitario(); //Não OK. MembroUniversitario não é só
                                                  //  Professor. Outros: Bolsista, TAs,...

//Um objeto da Subclasse SEMPRE pode ser atribuído para uma Superclasse.
//Um objeto da Superclasse NÃO PODE ser atribuido para uma Subclasse. Faltam campos.
```

//Classe Abstrata
//Exemplo 3: Sistema de cadastro de Bicicletas com Classe Abstrata. (Lista de Tipo Abstrato)

```
```

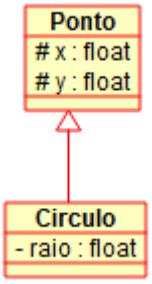
//Polimorfismo com Interfaces
//Exemplo 4: Sistema de cadastro de Bicicletas. (Lista de Tipo Abstrato)

```
public class Principal {

    public interface Bicicleta {
        //protected float aro;
        //public Bicicleta( float aro){
        //      this.aro = aro;
```

```
        //}
        public abstract float getAro();
        public abstract float cadastrar();
    }
    public class MontainBike implements Bicicleta {
        private float aro;
        public MontainBike(float aro){
            this.aro = aro;
        }
        public float getAro(){ return this.aro };
        public float cadastrar(){ return getAro(); }
    }
public class Speed implements Bicicleta {
        private float aro;
        public Speed(float aro){
            this.aro = aro;
        }
        public float getAro(){ return this.aro };
        public float cadastrar(){ return getAro(); }
    }
public class Ergonomica implements Bicicleta {
        private float aro;
        public Ergonomica(float aro){
            this.aro = aro;
        }
        public float getAro(){ return this.aro };
        public float cadastrar(){ return getAro(); }
    }

    public void iniciar(){
            ArrayList<Bicicleta>  lista = new ArrayList<>();

            Bicicleta bicicleta = new MontainBike( 15.0f );
            lista.add( bicicleta );

            bicicleta = new Speed( 20.0f );  //polimorfismo
            lista.add( bicicleta );

            bicicleta = new Ergonomica( 25.0f );   //polimorfismo
            lista.add( bicicleta );

            for( Bicicleta  b : lista  )
                System.out.println( b.cadastrar() );
    }

    public static void main( String [ ] args ){
            Principal principal = new Principal();
            principal.iniciar();
```

```
    }

}
```

//Exemplo 5: Formas



//Circulo eh um Ponto

```
public class Principal {

    public class Ponto {
        protected float x;
        protected float y;

        public Ponto(float x, float y){
            this.x = x;
            this.y = y;
        }
        public String toString(){ return “SOU UM PONTO: X: “ + x + “ Y: “ + y;  }
    }

    public class Circulo extends Ponto {
        private float raio;

        public Circulo( float x, float y, float raio ){
```

```
                super( x, y );
                this.raio = raio;
            }
            public String toString(){
               return “SOU UM CIRCULO: X: “ + x + “ Y: “ + y + “ Raio: “ + raio;
            }
    }

    public Principal(){

            Ponto p = new Circulo(1.0f, 2.0f, 3.0f);
            System.out.println( p );

            Circulo c = new Circulo( 1.0f, 2.0f, 3.0f);
            c = (Circulo) p;
            System.out.println( c );

            p = new Ponto( 4.0f, 5.0f );
            System.out.println( p );

    }

    public static void main(String [ ] args ){

             new Principal();
    }


}
```
