PROGRAMMAZIONE ORIENTATA AGLI OGGETTI
A.A. 2020/2021

Libro di testo:
C. Nigro, L. Nigro: **Programmazione Orientata agli Oggetti in Java**,
Pitagora Editrice (BO).

Libero Nigro

Concetti base

- Classi, oggetti, variabili di istanza, metodi (costruttori, accessori, mutatori, etc.). Il metodo toString.
- Classi di oggetti mutabili e immutabili.
- Primi esempi sviluppati in una directory es. c:\primi_programmi.
- Strumenti textpad/notepad, javac/java a riga di comando.
- Un esempio di classe di oggetti dotati di *stato mutabile*: una classe Punto nel piano cartesiano, per applicazioni geometriche. Un punto si può spostare etc.
- Il pronome this
- Sviluppo Java completo.

Una classe Punto

```
public class Punto{
  private double x, y; //variabili di istanza o campi dell'oggetto
  public Punto(){//costruttore di default
    x=0; y=0;
  }
  public Punto( double x_, double y_ ){//costruttore normale
    x=x_; y=y_;
  }
  public Punto( Punto p ){//costruttore di copia
    x=p.x; y=p.y;
  }
  public double getX(){ return x; } //metodi getter
  public double getY(){ return y; }
  public void sposta( double x_, double y_){//metodo mutatore
    x=x_; y=y_;
  }//sposta
  public double distanza( Punto p ){  return Math.sqrt((p.x-x)*(p.x-x)+(p.y-y)*(p.y-y));  }//distanza
```

Classe Punto (continuazione)

```
  public String toString(){
    return "Punto("+x+","+y+“)";
  }//toString

  public static void main( String[] args ){
     Punto p=new Punto(5,7);
     System.out.println(“x=“+p.getX()+” y=“+p.getY()); //System.out.println(p);
     p.sposta(4,-6);   System.out.println(p);
     Punto q=new Punto();
     double d=q.distanza(p); //notazione OO: ricevitore.nome_metodo(parametri)
     System.out.println(“distanza =“+d);
  }//main

}//Punto
```

Compilazione/esecuzione a riga di comando

C:\primi_programmi>javac Punto.java

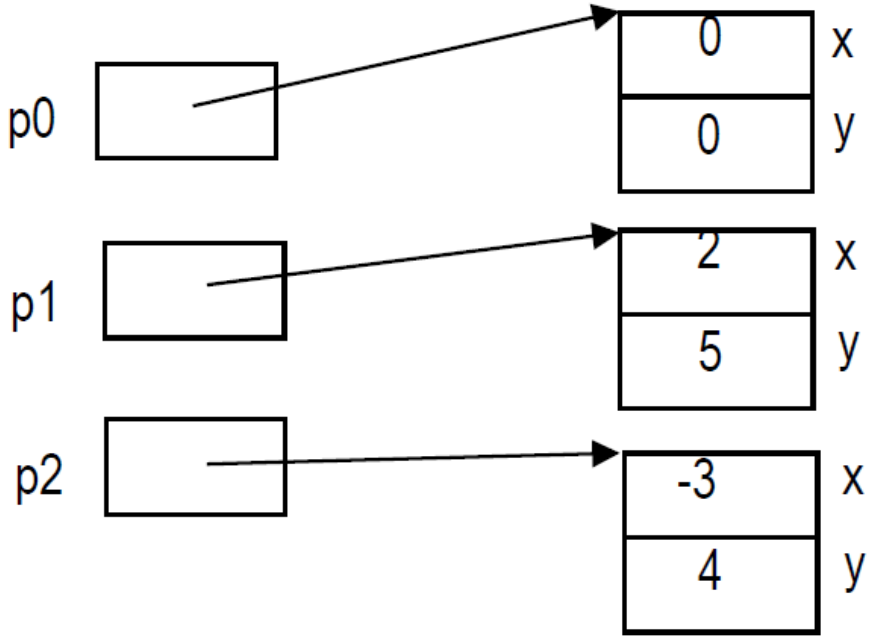
C:\primi_programmi>java Punto.java (in quanto dotato di un main demo)

Si assume un'installazione del JDK di Java da 8 in su (es. 15, uscita a settembre 2020)

Approfondimenti

- In un main si possono dichiarare e creare tre punti p0, p1 e p2:

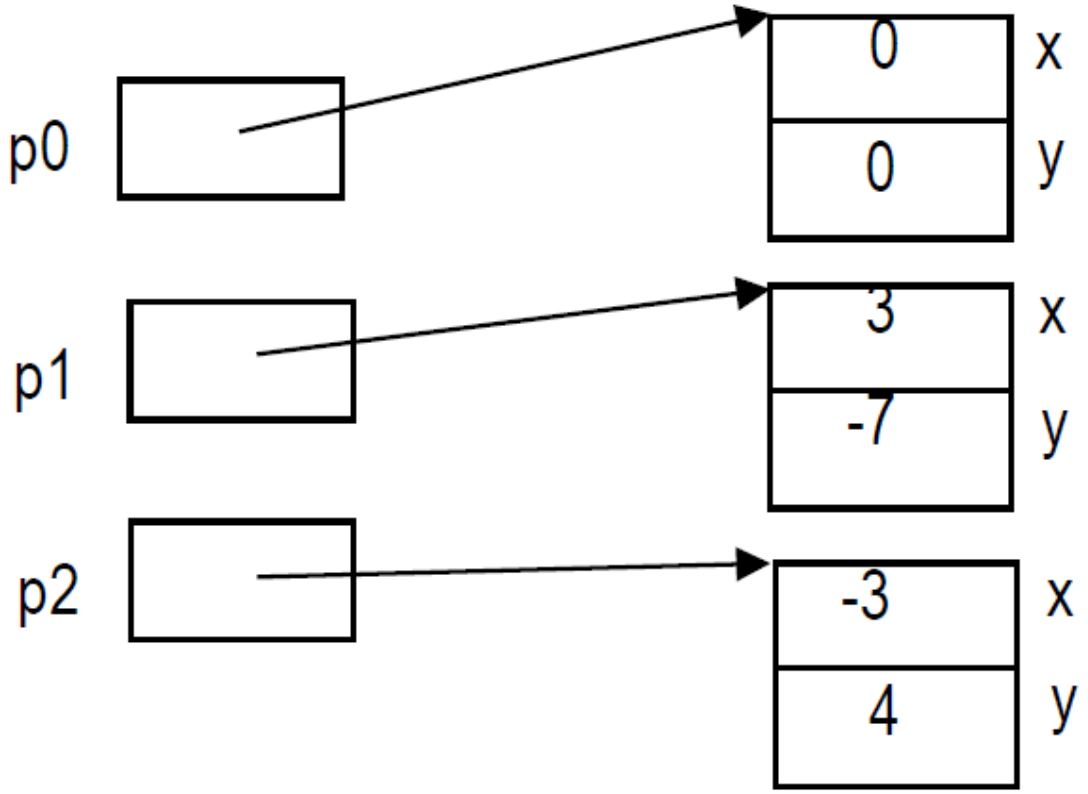
```
Punto p0=new Punto();
Punto p1=new Punto(2,5);
Punto p2=new Punto(-3,4);
```



Modello di memoria che chiarisce il senso Java che:
un <u>oggetto è un riferimento</u>

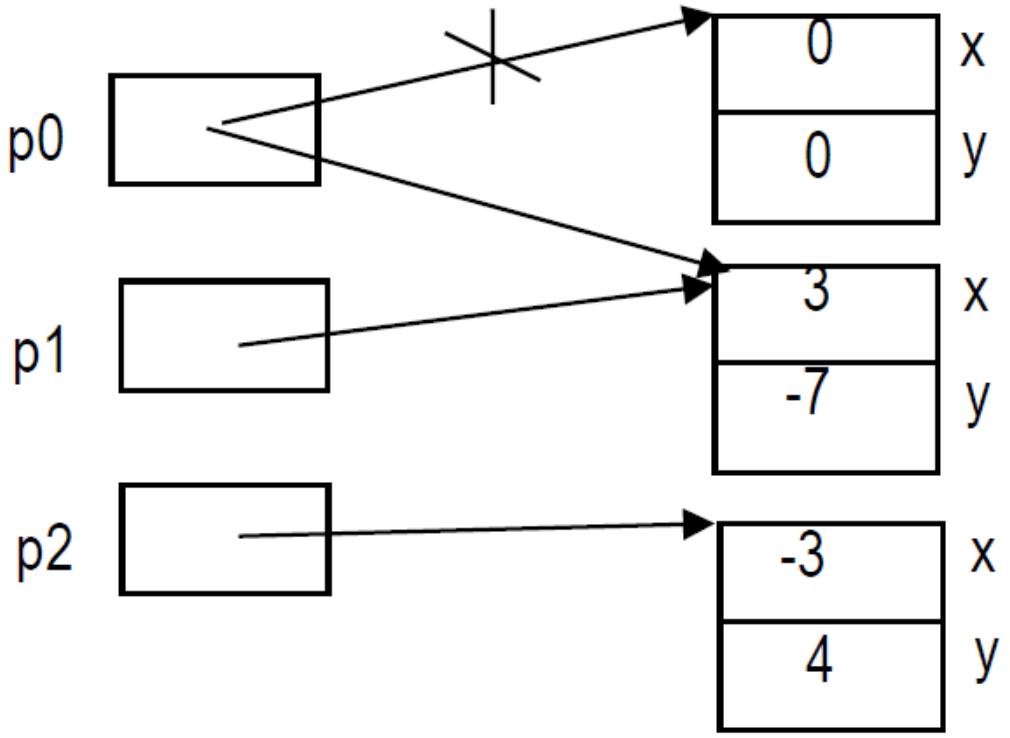
Approfondimenti

p1.sposta(3,-7);



Approfondimenti – assegnazione tra oggetti

p0=p1;



Il precedente oggetto puntato da p0 è ora inaccessibile ed è diventato **garbage**

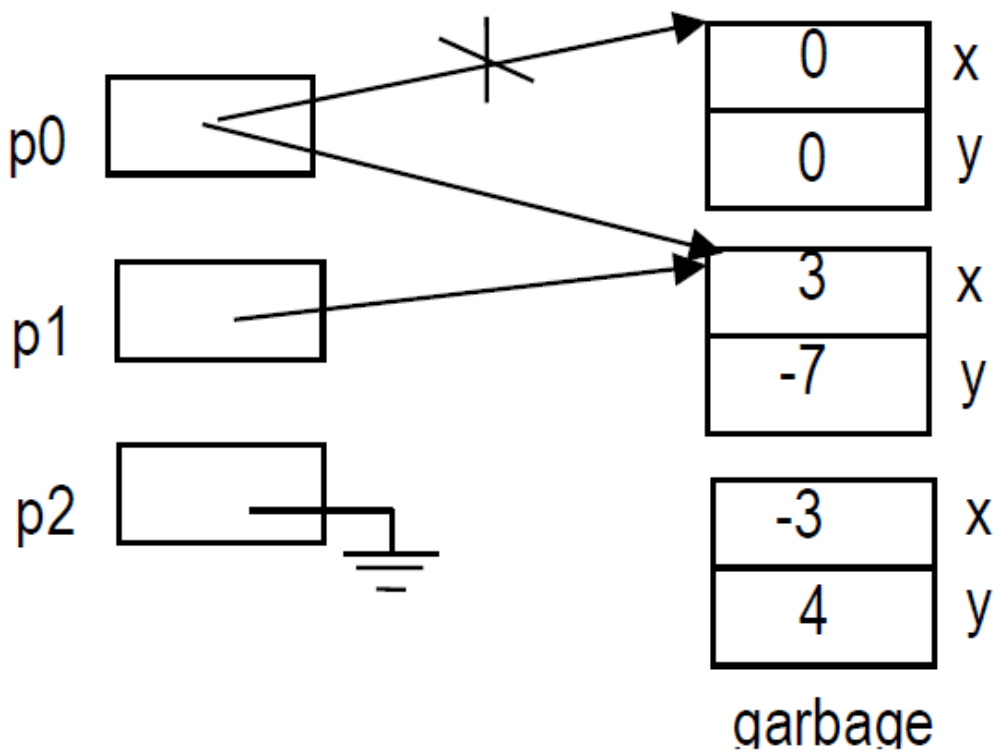
p0 e p1 sono ora in **aliasing**:

```
p0.sposta( 10, 20 );
System.out.println( p1 );
```

Si stampa: Punto(10,20)

La costante null (assenza di riferimento)

p2=null;



Approfondimenti

- All'interno di un metodo non è noto il nome dell'oggetto ricevitore:

```
ricevitore.nome_metodo( parametri );
p1.getX();
```

- Tuttavia è possibile usare il pronome **this** per riferirsi a «questo» oggetto, sul quale il metodo viene invocato
- **Dettaglio implementativo**: Java passa (tacitamente) al metodo un ulteriore parametro che reca il riferimento all'oggetto this (ricevitore, p1).
- Quando si usa this, si usa questo parametro addizionale, implicito.

Effetti di this

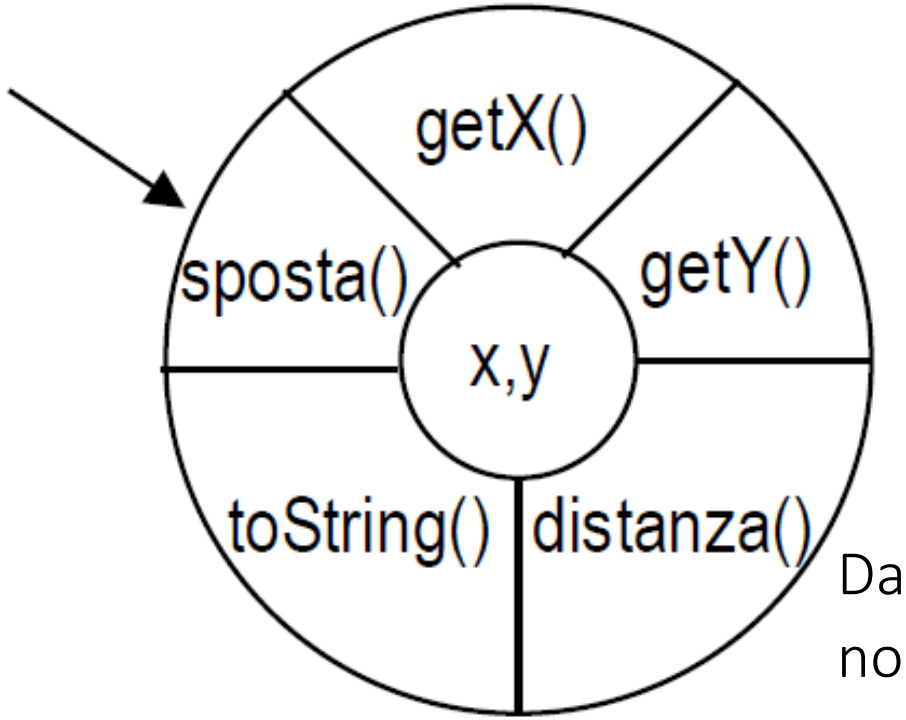
```
public Punto(){
    this(0,0);
}

public Punto( double x, double y ){ //non serve battezzare nomi diversi per i parametri
    this.x=x; this.y=y;
}

public void sposta( double x, double y ){
    this.x=x; this.y=y;
}

public double distanza( Punto p ){
    return Math.sqrt((p.x-this.x)*(p.x-this.x)+(p.y-this.y)*(p.y-this.y)); //CHIAREZZA
 }
```

Oggetti incapsulati e privatezza dello stato



Da **fuori** della classe Punto
non è possibile accedere ai campi
x ed y di un oggetto Punto
Bisogna usare i metodi getX() e getY()