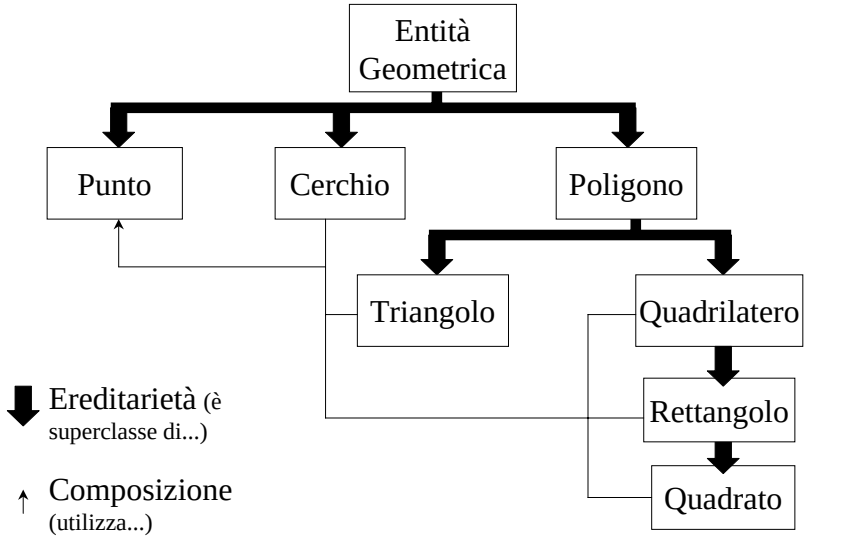
Esercitazione n° 3

Obiettivi:

- Capacità di ***analisi*** e di ***estensione*** di progetti esistenti
- Linguaggio Java:
  - ***Ereditarietà delle classi***
  - Utilizzo di ***costruttori e metodi*** di ***superclasse***
  - ***Classi astratte***
  - Visibilità: ***private, public, package, protected***
- Esempio guida: modellizzazione di ***figure geometriche*** (punto, cerchio, poligono, triangolo,...)

Il package Esercitazione3



La classe `EntitaGeometrica`

```
public abstract class EntitaGeometrica
      extends Object {

  public abstract void muovi(Punto vettore);

  public void disegna() {
  System.out.println("Disegno la figura..."); }

  public void stampa() {
  System.out.println("Sono una generica figura
   geometrica"); }
}
```


La classe `Punto` (1)

```
public class Punto extends EntitaGeometrica {

  private double x, y;

  public Punto() { x = y = 0; }

  public Punto(double newx, double newy) {
  x = newx; y = newy; }

  public double getx() { ... }
  public double setx(double newx) { ... }
  public double gety() { ... }
  public double sety(double newy) { ... }
```

La classe **Punto** (2)

```
public boolean equals(Punto p) {
  if ((x==p.x)&&(y==p.y))
    return true;
  return false; }

public void sommaVett(Punto vett) {
  x = x + vett.getx();
  y = y + vett.gety(); }

public void muovi(Punto nuovoCentro) {
  x = nuovoCentro.getx();
  y = nuovoCentro.gety(); }
```


La classe **Cerchio**: i costruttori

```
public class Cerchio extends EntitaGeometrica {

  private Punto centro;
  private double raggio;

  public Cerchio(Punto c, double r) {
  centro = new Punto(c.getx(), c.gety());
  raggio = r; }

  public Cerchio(Punto c) {
  centro = new Punto(c.getx(), c.gety());
  raggio = 1; }

  public Cerchio() {
  centro = new Punto();
  raggio = 1; }
```


La classe **Cerchio**: i metodi

```
public void muovi(Punto vettore) {
  centro.sommaVett(vettore); }

public boolean equals(Cerchio cerchio) {
  if ((centro.equals(cerchio.centro))&&
       (raggio==cerchio.raggio)) return true;
  else return false; }

public double circonferenza() {
  return (2*raggio*Math.PI); }

public double area() {
  return (raggio*raggio*Math.PI); }

public void stampa() { ... }
```


La classe **Poligono**

```
public abstract class Poligono extends EntitaGeometrica {

  protected double lato(Punto a, Punto b) {
       double temp = Math.sqrt((b.gety()-a.gety())*
              (b.gety()-a.gety())+(b.getx()-a.getx())*
              (b.getx()-a.getx()));
       return temp; }

  public abstract double perimetro();

  public abstract double area();

  public void stampa() {
       System.out.println("\nSono un poligono generico");
   }
}
```

La classe **Triangolo**: i costruttori

```
public class Triangolo extends Poligono {

  private Punto v1, v2, v3;

  public Triangolo(Punto a, Punto b, Punto c) {
  v1 = new Punto(a.getx(), a.gety());
  v2 = new Punto(b.getx(), b.gety());
  v3 = new Punto(c.getx(), c.gety()); }

  public Triangolo() {
  v1 = new Punto();
  v2 = new Punto();
  v3 = new Punto(); }
```


La classe **Triangolo**: i metodi

```
public void muovi(Punto vettore) {
  v1.sommaVett(vettore);   v2.sommaVett(vettore);
  v3.sommaVett(vettore); }

public boolean equals(Triangolo t) {
  if (v1.equals(t.v1)&&v2.equals(t.v2)
       &&v3.equals(t.v3)) return true;
  else return false; }

public double perimetro() {
  double lato1 = lato(v1,v2); double lato2 = lato(v2,v3);
  double lato3 = lato(v3,v1);
  return (lato1 + lato2 + lato3); }

public double area() { return 0; }
public void stampa() { … }
```


La classe **Quadrilatero**: i costruttori

```
public class Quadrilatero extends Poligono {

  protected Punto v1, v2, v3, v4;

  public Quadrilatero(Punto a, Punto b, Punto c,
       Punto d) {
       v1 = new Punto(a.getx(), a.gety());
       v2 = new Punto(b.getx(), b.gety());
       v3 = new Punto(c.getx(), c.gety());
       v4 = new Punto(d.getx(), d.gety()); }

  public Quadrilatero() {
       v1 = new Punto();  v2 = new Punto();
       v3 = new Punto();  v4 = new Punto(); }
```


La classe **Quadrilatero**: i metodi

```
public void muovi(Punto vettore) { ... }

public boolean equals(Quadrilatero q) {
  if (v1.equals(q.v1)&&v2.equals(q.v2)&&
       v3.equals(q.v3)&&v4.equals(q.v4)) return true;
  else return false; }

public double perimetro() {
  double lato1 = lato(v1,v2); double lato2 = lato(v2,v3);
  double lato3 = lato(v3,v4); double lato4 = lato(v4,v1);
  return (lato1 + lato2 + lato3 + lato4); }

public double area() { return 0; }

public void stampa() { ... }
```

Da fare in Lab: la classe **Rettangolo**

```
public class Rettangolo extends Quadrilatero {
  public Rettangolo(Punto a, Punto b, Punto c, Punto d)
  public Rettangolo(Punto a, Punto b)
  public Rettangolo()
  public void muovi(Punto vettore)
  public boolean equals(Rettangolo r)
  public double perimetro()
  public double area()
  public void stampa()
```


Da fare in lab: la classe **Quadrato**

```
public class Quadrato extends Rettangolo {
  public Quadrato(Punto a, Punto b, Punto c, Punto d)
  public Quadrato(Punto a, int lato)
  public Quadrato()
  public void muovi(Punto vettore)
  public boolean equals(Quadrato q)
  public double perimetro()
  public double area()
  public void stampa()
```