Linguaggi di Programmazione

a.a. 13/14

docente: Gabriele Fici

gabriele.fici@unipa.it

8 - Classi astratte e Interfacce

8 - Classi astratte e Interfacce

- Java consente di definire classi in cui uno o più metodi non sono implementati, ma solo dichiarati (metodi `abstract`)
- Questi metodi non hanno un corpo tra parentesi graffe ma solo la dichiarazione terminata con `;`
- Una classe che ha almeno un metodo astratto si dice classe astratta, e va anch'essa marcata come `abstract`

Esempio:

```
public abstract class Poligono {
    public abstract double area();
    public abstract double perimetro();
}// classe totalmente astratta
 // i metodi vanno implementati nelle sottoclassi
```

8 - Classi astratte e Interfacce

- Una classe astratta *non può* essere istanziata
- Serve come superclasse comune per creare una gerarchia di ereditarietà
- Nelle sottoclassi si possono implementare i metodi astratti

8 - Classi astratte e Interfacce

```
public abstract class Poligono {
    public abstract double area();
    public abstract double perimetro();
}
```

```
public class Pentagono extends Poligono {
  public static final double NF = 6.88;
  protected double lato;
  public Pentagono (double lato){
     this.lato = lato;}
  public double perimetro(){
     return 5 * lato;}
  // ora il metodo perimetro è concreto
  public double area(){
     return perimetro * NF / 2;}
  // ora il metodo area è concreto
}
```

8 - Classi astratte e Interfacce

```
public abstract class Quadrilatero extends Poligono{
  protected int a, b, c, d;
  public Quadrilatero (int a, int b, int c, int d){
       this.a=a; this.b=b; this.c=c; this.d=d;
  }
  public double perimetro(){
       return a + b + c + d;
  } // ora il metodo perimetro è concreto
}
```

```
public class Rettangolo extends Quadrilatero {
  public Rettangolo (int base, int altezza){
       super (base, altezza, base, altezza);
  }
  public double area(){
       return a * b;
  } // ora il metodo area è concreto
}
```

8 - Classi astratte e Interfacce

- Le classi astratte permettono di organizzare una gerarchia di ereditarietà singola
- Ma se si vuole imporre l'implementazione di una lista di metodi indipendenti dalla gerarchia di ereditarietà?
- Ad esempio, si vuole definire una serie di metodi che permettano di disegnare una figura
- Per questo scopo, si usano le interfacce

8 - Classi astratte e Interfacce

- Un'*interfaccia* è una collezione di dichiarazioni di metodi non implementati
- Può contenere attributi solo se costanti (che vengono settati automaticamente `static`, `final` e `public`)
- Si dichiara con `interface` al posto di `class`
- Al contrario di una classe astratta, in un'interfaccia *tutti* i metodi devono essere privi di implementazione

Esempio:

```
public interface Disegnabile {
   public void setColore (int c);
   public void setSpessore (int s);
   public void disegna ();
}
```

8 - Classi astratte e Interfacce

- Una classe può implementare anche più di un'interfaccia
- Se una classe implementa un'interfaccia, deve fornire un'implementazione di *tutti* i metodi dell'interfaccia
- Perciò, un'interfaccia è un contratto tra chi la definisce e chi la implementa, perché garantisce la fornitura di certi servizi
- Per implementare un'interfaccia si usa la parola chiave `implements` seguita dall'elenco delle interfacce da implementare

8 - Classi astratte e Interfacce

Esempio:

```
public class RettangoloDisegnabile extends Rettangolo
implements Disegnabile {
    protected int colore, spessore;

    public void setColore(int c){this.colore = c;}
    public void setSpessore(int s){this.spessore = s;}
    public void disegna (){...}
}
```

8 - Classi astratte e Interfacce

- E' possibile dichiarare un riferimento ad un'interfaccia
- Questo può contenere una variabile avente come tipo una qualsiasi classe che implementa l'interfaccia

Esempio:

```
Disegnabile d;
d = new RettangoloDisegnabile(5,6);
```

8 - Classi astratte e Interfacce

- Si può allora sfruttare il polimorfismo per creare, ad esempio, un array di riferimenti a interfacce e richiamare i metodi propri di ciascuna classe

Esempio:

```
Disegnabile[] figure = new Disegnabile[2];
figure[0] = new RettangoloDisegnabile(5,6);
figure[1] = new PentagonoDisegnabile(7);

for ( int i = 0 ; i < figure.length ; i++ ){
  figure[i].setSpessore(1);
  figure[i].disegna();
}
```

8 - Classi astratte e Interfacce

- Il fatto che l'interfaccia sia indipendente dalla gerarchia di ereditarietà permette di usarla anche per classi che non sono in relazione di ereditarietà tra loro
- Ad esempio, si potrebbe definire una classe `TestoDisegnabile` che implementa l'interfaccia `Disegnabile`
- Il polimorfismo permette quindi di creare array di riferimenti a interfacce che contengono come variabili istanze di classi molto eterogenee