Programmazione ad oggetti vs procedurale

Programmazione imperativa

Possiamo programmare utilizzando i seguenti tipi di dati:

- Tipi di dato primitivi (int, double, char, boolean, ecc...)
- Le stringhe
- Gli array

I programmi consistono di una sequenza di comandi, strutture di controllo (cicli, scelte condizionali, ecc...) ed eventualmente metodi ausiliari che consentono di manipolare i dati per calcolare il risultato voluto.

Questo modo di programmare prende il nome di **PROGRAMMAZIONE IMPERATIVA**, imperativa in quanto basata su comandi

Programmazione imperativa

Nella programmazione imperativa:

- Un programma prevede uno stato globale costituito dai valori delle sue variabili
- I comandi del programma modificano lo stato fino a raggiungere uno stato finale (che include il risultato)

Programmazione imperativa

Ad esempio, il seguente programma (che calcola il prodotto di x e y) ha la seguente dinamica:

```
int x=10, y=3, p=0;
for (int i=0; i<y; i++)
p+=x;
```

Procedurale vs OOP

- Nella programmazione procedurale, il codice è centrale e i dati sono subordinati
- abbiamo programmi che agiscono sui dati che di solito non sono strettamente collegati
- Nella programmazione a oggetti, gli oggetti sono l'elemento centrale.
- Un oggetto consiste nei **dati** (attributi, proprietà, ...)
- e nel codice che opera su tali dati: **metodi**
- **dati e metodi** sono strettamente collegati: è il concetto di **incapsulamento**, che
- l'**incapsulamento** permette anche di nascondere l'implementazione interna, utilizzando l'oggetto attraverso l'**interfaccia** pubblica.

Per esempio,

abbiamo un numero e vogliamo raddoppiarlo.

Nel modo procedurale faremmo:

```
n = n * 2
```

Il codice moltiplica n per 2 e registra il risultato in n.

Nella programmmazione orientata agli oggetti

si invia un "messaggio" all'oggetto chiamando un metodo per raddoppiare:

```
n.double();
```

puoi creare un oggetto di tipo stringa che accetta la chiamata al metodo `double()`, ma lo implementa in maniera differente.

```
class Operazioni{


        public int double(int n){
                return n * 2;
        }

        public String double(String s){
                return s+s;
        }

}
```

Il vantaggio di questa tecnica è definito **polimorfismo**.

Se il programma richiede di replicare la procedura su un oggetto di tipo string come "bob", nel modo procedurale occorre invocare una nuova funzione con un codice e un nome differente.

OOP