


Introduzione alla Programmazione Orientata agli Oggetti (OOP)

Tutti i linguaggi di programmazione prevedono l'astrazione, infatti non sono altro che un'astrazione dei linguaggi assemblativi che, a loro volta, sono astrazioni.

 C per esempio, è un'astrazione del linguaggio assembly e come tutti i linguaggi imperativi ha implementato un'astrazione che ti richiede di **pensare in termini di struttura del computer** invece che di **struttura del problema**.

Durante gli anni '60 i linguaggi già possedevano astrazione rispetto ai registri e avevano un uso avanzato della memoria.

Con l'unico problema che **erano fortemente legati alle stesse modalità di organizzazione del flusso di operazioni**, l'unico meccanismo di salto incondizionato era il GO TO.

E già alla fine degli anni '60 cominciarono a nascere le **prime critiche** verso la programmazione tradizionale.

Questo tipo di programmazione venne sostituita con la **Programmazione strutturata** che si basa principalmente sul **Teorema di Bohm-Jacopini**.

Teorema di Bohm-Jacopini

Qualunque algoritmo può essere implementato utilizzando tre sole strutture, la sequenza, la selezione ed il ciclo (iterazione), da applicare ricorsivamente alla composizione di istruzioni elementari.

Andando a vedere nello specifico queste operazioni:

- **Sequenza:** Esecuzione ordinata di uno statement.
- **Selezione:** Uno statement viene selezionato a seconda dello stato del programma.
- **Ciclo:** Uno statement eseguito fino a che il programma non raggiunge un determinato stato.

Inoltre in questo tipo di programmazione dobbiamo rispettare i seguenti requisiti:

- **Completezza:** Le tre strutture devono avere almeno un **rappresentazione sintattica**.
- **Singolo punto di ingresso e uscita:** All'interno di una struttura si deve poter indicare un solo **punto di ingresso e uscita**.

- **Compatibilità:** Ogni struttura di controllo deve poter essere considerata un **macro-statement** in modo da poter essere usata ricorsivamente.

Con il passare degli anni venne inventata la **Programmazione Procedurale** che si basa appunto su *procedure*, cioè blocchi di codice racchiusi da delimitatori e identificati da un nome richiamabili in qualunque punto del programma.

Tali blocchi assumono il nome di:

- *Subroutine*
- *Procedure*
- *Funzioni*
- *Metodi*

Che stanno alla base della **Programmazione Orientata agli Oggetti**.

La programmazione orientata ad oggetti ci permette di **approcciarci al problema utilizzando come strumenti gli elementi presenti nel problema**.

Esistono 5 caratteristiche basilari su cui si basano i linguaggi Orientati agli Oggetti:

1. **Tutto è un oggetto:** Possiamo pensare ad un oggetto come ad una **variabile**, infatti esso può contenere dati e possiamo **fargli delle richieste** per fare delle operazioni:

```
public class Main{
    int x = 5;

    public static void main(String[] args){
        Main oggetto = new Main();//Oggetto con valore x=5
        System.out.println(oggetto.x);//Chiedi oggetto il valore x
    }
}
```

2. **Un programma è un insieme di oggetti che si dicono a vicenda cosa fare:** Gli oggetti si mandano dei messaggi per funzionare, cioè delle richieste che chiamano dei **metodi**.
3. **Ogni oggetto ha una sua memoria composta da altri oggetti.**
4. **Ogni oggetto ha un tipo:** Ogni oggetto è un *istanza di una classe*, una classe può essere visto come sinonimo di tipo.
5. **Tutti gli oggetti di un particolare tipo possono ricevere gli stessi messaggi.**

Quindi ogni oggetto è come se fosse un *mini-computer* poiché ha:

- **Stato**

- Comportamenti
- Identità

La particolarità degli oggetti è che nonostante siano essenzialmente diversi tra loro fanno tutti parte di una **classe** di oggetti in comune.

Quindi una classe **descrive un insieme di oggetti che hanno caratteristiche (dati) e comportamenti (funzioni) identiche.**

Possiamo pensare agli oggetti come delle **interfacce** che permettono di **semplificare l'utilizzo dell'applicazione** che stiamo scrivendo e come dei **servizi** perché **permettono all'utente di ottenere qualcosa** dai servizi offerti dagli oggetti.

 **Generalmente possiamo vedere gli oggetti come delle istanze delle classi:**

Esempio:

Classe: Persona

Oggetto: Mario, Luigi, Marco

I quattro principi fondamentali che caratterizza la OOP sono:

- **Encapsulation:** La capacità di **nascondere dettagli implementativi** e quindi poter accedere ad informazioni solo tramite *accessor e mutators*.
- **Abstraction:** Correlata alla precedente, ha a che fare con la **definizione di modelli per gli oggetti che vogliamo rappresentare.**
- **Inheritance:** Permette di raccogliere a fattor **comune le caratteristiche condivise da oggetti simili** definendo modelli che rappresentino tali caratteristiche condivise da oggetti simili.
- **Polymorphism:** **Abilità degli oggetti di prendere diverse forme,** in altre parole permette ad oggetti appartenenti alle stesse classi di rispondere diversamente agli stessi messaggi.