**Principi base della OOP**

1. ***ASTRAZIONE***
2. ***INCAPSULAMENTO***
3. ***EREDITARIETÀ***
4. ***POLIMORFISMO***

**ASTRAZIONE**

**Generalizzare**, in modo tale da avere delle interfacce comuni a tutti gli oggetti di quel tipo.

**INCAPSULAMENTO (INFORMATION HIDING)**

Modificatori di accesso(**public, private & protected**).  
  
**EREDITARIETÀ**

L’ereditarietà è un meccanismo di **astrazione gerarchica** che consente di derivare classi da altre classi, in modo da “conservare“ le proprietà della classe base nelle classi derivate.  
  
**POLIMORFISMO**

Funzioni che hanno lo stesso nome ma assumono comportamenti diversi a seconda del chiamante   
Es. Classe (**PERSONA**) che ha un metodo ***print*** e da questa classe derivo (**STUDENTE**) e (**LAVORATORE**) all’interno delle quali definisco la funzione print, **LA FUNZIONE PRINT DIVENTA UNA** **FUNZIONE POLIMORFA**.

**CLASSE**

* DATI
* FUNZIONI

Codesti sono dei membri della classe che vengono chiamati **DATI** 🡪 **ATTRIBUTI**, FUNZIONI 🡪 **METODI**

|  |  |
| --- | --- |
| CLASSE | STRUCT |
| Forniscono la possibilità di modificare i livelli di accesso ai membri della classe stessa. Default: PRIVATE | Tutti i dati e le funzioni definite all’interno della struct, sono **PUBLIC** (sono accessibili pubblicamente da chiunque).  Default: **PUBLIC** |

**CLASSE** **vs OGGETTO**

**Costruttore:**

* **Stesso nome della classe;**
* **Nessun tipo di ritorno (Nemmeno VOID);**
* **Responsabile dell’inizializzazione dei valori degli attributi.**

La classe ha bisogno di un costruttore per poter essere stanziata:

* **Default**
  + Costruttore senza parametri**.**
* **Completo**
* **Copy Constructor**

class B : public A private A 🡪 B private  
protected A 🡪 B protected  
public A 🡪 B public

class B : private A private A 🡪 B private  
protected A 🡪 B private  
public A 🡪 B private

class B : protected A private A 🡪 B private  
protected A 🡪 B protected  
public A 🡪 B protected