UNIVERSITÀ DELLA CALABRIA

DIPARTIMENTO DI **MATEMATICA E INFORMATICA**

Programmazione ad Oggetti

Carmine Dodaro

Anno Accademico 2019/2020

Programmazione ad oggetti

Cos'è

È un paradigma di programmazione basato sul concetto di oggetto. Un oggetto può contenere dati, di solito chiamati attributi, e codice sotto forma di metodi.

I concetti fondamentali

- Incapsulamento
- Ereditarietà
- Polimorfismo
- Composizione

Oggetti e classi

Cos'è un oggetto?

Un oggetto tipicamente è definito da due componenti: gli attributi e i comportamenti.
Ad esempio, nel definire un'automobile possiamo pensare ad attributi come la cilindrata, velocità massima, colore, ecc. e a comportamenti come accelerare, frenare, ecc.

Quindi, nella definizione più generale, un oggetto è un'entità che contiene attributi e comportamenti.

Classe vs oggetto

Una classe è come se fosse il progetto dell'oggetto. La classe rappresenta la descrizione astratta di un oggetto, ad esempio la descrizione degli attributi e dei metodi, mentre l'oggetto rappresenta un'istanziazione specifica della classe.

Esempio

Prodotto.h

```
#ifndef PRODOTTO_H
#define PRODOTTO_H

class Prodotto {
  public:
    Prodotto();
    int getId() const;
    void setId(int);
    float getPrezzo() const;
    void setPrezzo(float);

  private:
    int id;
    float prezzo;
};

#endif
```

La classe Prodotto descrive gli attributi e i metodi.

Prodotto.cpp

```
#include "Prodotto.h"

Prodotto::Prodotto() {
  id = 0;
  prezzo = 0.0;
}

int Prodotto::getId() const {
  return id;
}

void Prodotto::setId(int i) {
  id = i;
}

float Prodotto::getPrezzo() const {
  return prezzo;
}

void Prodotto::setPrezzo(float p) {
  prezzo = p;
}
```

Esempio

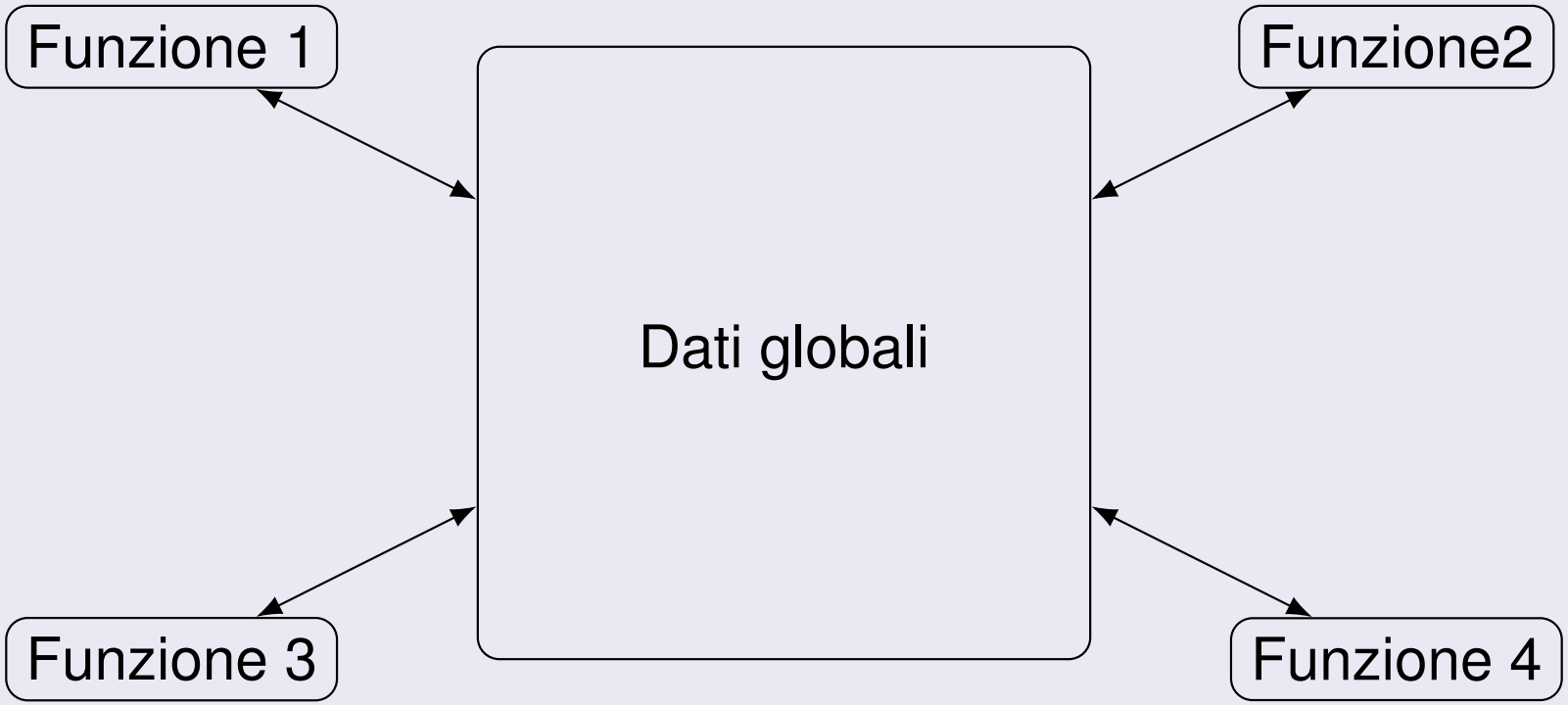
main.cpp

```
#include "Prodotto.h"
#include <iostream>
using namespace std;

int main() {
  Prodotto p1;
  p1.setId(1);
  p1.setPrezzo(3.5);

  cout << "Prodotto " << p1.getId() << " " << p1.getPrezzo() <<
      endl;
  return 0;
}
```

L'oggetto p1 rappresenta un'istanziazione della classe Prodotto.

Programmazione strutturata vs ad oggetti

Programmazione strutturata



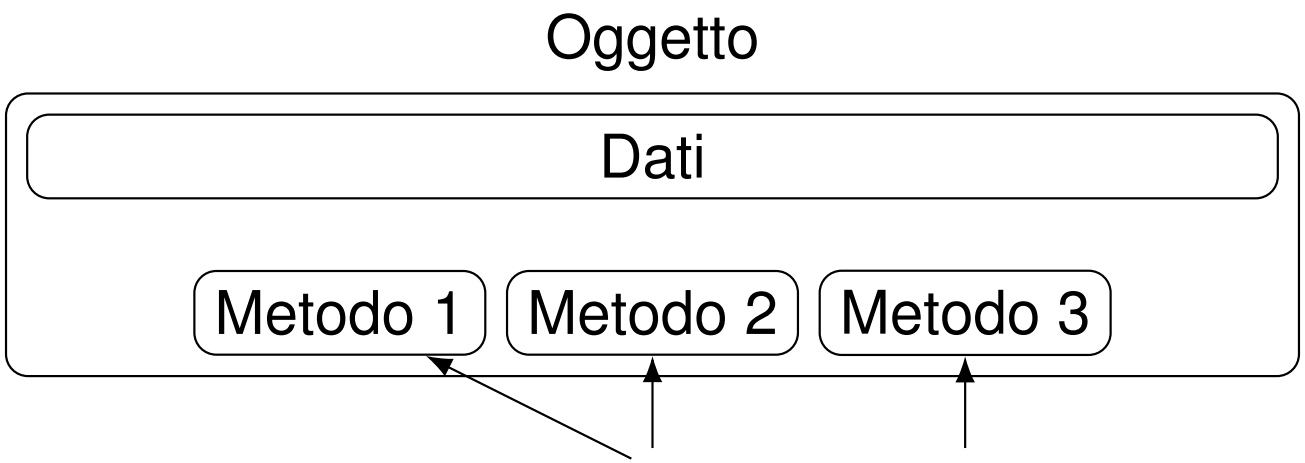
- I dati sono separati dalle funzioni e, spesso, sono globali
- L'accesso ai dati non è quindi controllato e prevedibile, perché funzioni diverse accedono allo stesso dato
- Inoltre, la mancanza di controllo su chi ha accesso al dato rende più difficile capire trovare errori

Programmazione strutturata vs ad oggetti

Programmazione ad oggetti

Nella programmazione ad oggetti i dati e i comportamenti sono contenuti all'interno di un oggetto, che può prevedere dei meccanismi per impedire l'accesso agli attributi e ai metodi all'esterno (data hiding).

Combinando attributi e metodi all'interno della stessa entità (incapsulamento) abbiamo la possibilità di controllare l'accesso ai dati e ai metodi dell'oggetto.



Incapsulamento

Cosa fare

Nella progettazione di una classe è importante tenere presente che un oggetto dovrebbe consentire di utilizzare solo quei metodi necessari per l'interazione con l'oggetto. I dettagli che non sono pertinenti per l'uso dell'oggetto dovrebbero essere nascosti agli altri oggetti.

Ad esempio, un oggetto Calcolatrice per sommare e moltiplicare due numeri dovrebbe esporre solo i metodi per effettuare queste due operazioni. Gli attributi e gli algoritmi usati non devono essere visibili all'utilizzatore dell'oggetto.

Getters e setters

I getters e i setters supportano il concetto di data hiding. Infatti, il codice esterno alla classe non può accedere direttamente al contenuto dei dati, ma devono interagire con i getters e i setters.

Progettazione di una classe

Regole generali

- È consigliabile offrire un'interfaccia pubblica più semplice e intuitiva possibile
- Chi usa la classe non dovrebbe conoscere i dettagli implementativi della classe
- Le classi devono essere progettate tenendo a mente le esigenze di chi la userà