UNIVERSITÀ PARTHENOPE
DiST
Dipartimento di Scienze e Tecnologie

Programmazione II e Laboratorio di P2

Introduzione
a
Classi e Oggetti

Angelo Ciaramella

Introduzione

- Classe
  - meccanismo che consente di definire nuovi tipi di dato a partire da quelli esistenti
  - permette l'*information hiding*
  - generalizzazione del concetto di `struct`
  - `namespace` contenente i suoi membri

Fondamento

- Classe
  - set di membri
    - membri dati (campi)
    - membri funzioni (metodi)
      - inizializzazione (creazione)
      - copia
      - spostamento
      - pulizia (distruzione)
  - i membri sono accessibili tramite . o ->
  - è possibile definire operatori come +, ! e [] per una classe
  - è possibile associare degli specificatori ai metodi
    - public, protected, private
  - una struct è una class in cui i membri sono public per default

Struct Data

```
struct Date { // rappresentazione
int d, m, y;
};

void init_date(Data& d, int, int, int); // inizializzazione
void add_year(Data& d, int n); // aggiunge n anni a d
void add_month(Data& d, int n); // aggiunge n mesi a d
void add_day(Data& d, int n); // aggiunge n giorni a d
```

Esempio di Struct e funzioni associate senza connessione esplicita

Struct Date

```
struct Date { // rappresentazione
  int d, m, y;

  void init_date(int dd, int mm, int yy);
  // inizializzazione
  void add_year(int n); // aggiunge n anni a d
  void add_month(int n); // aggiunge n mesi a d
  void add_day(int n); // aggiunge n giorni a d
};
```

Esempio di Struct e funzioni membro associate con connessione esplicita

Struct Date

```
Date my_birthday;

void f()
{
  Date today;
  today.init_date(16,10,1996);
  my_birthday.init_date(30,12,1950);

  Date tomorrow = today;
  tomorrow.add_day(1);

// ...
}
```

Esempio di utilizzo della Struct Date

class Data

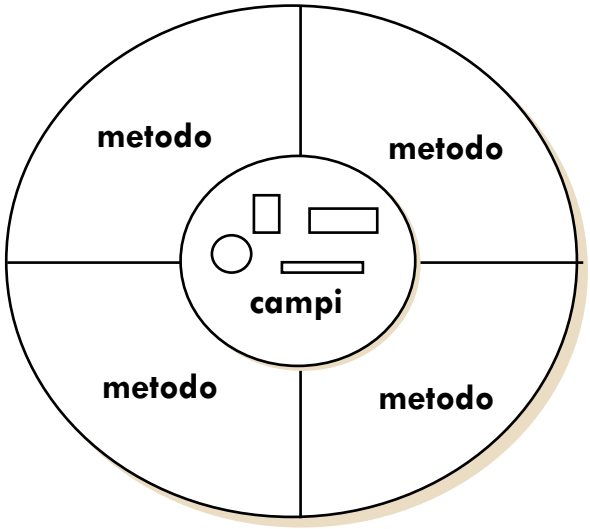
```
class Date { // rappresentazione
  private:
    int d, m, y;

  public:
    void init_date(int dd, int mm, int yy);
    // inizializzazione
    void add_year(int n); // aggiunge n anni a d
    void add_month(int n); // aggiunge n mesi a d
    void add_day(int n); // aggiunge n giorni a d
};
```

`private` per default

Esempio di class Date e information hiding

Classe

- Una classe ha
  - un nome
  - contiene due tipi di membri
    - campi e metodi



Tipo di dati astratti

Costruttori

```
class Date {
  int d, m, y;

  public:
   Date(int dd, int mm, int yy); // costruttore
   // ...
};
```

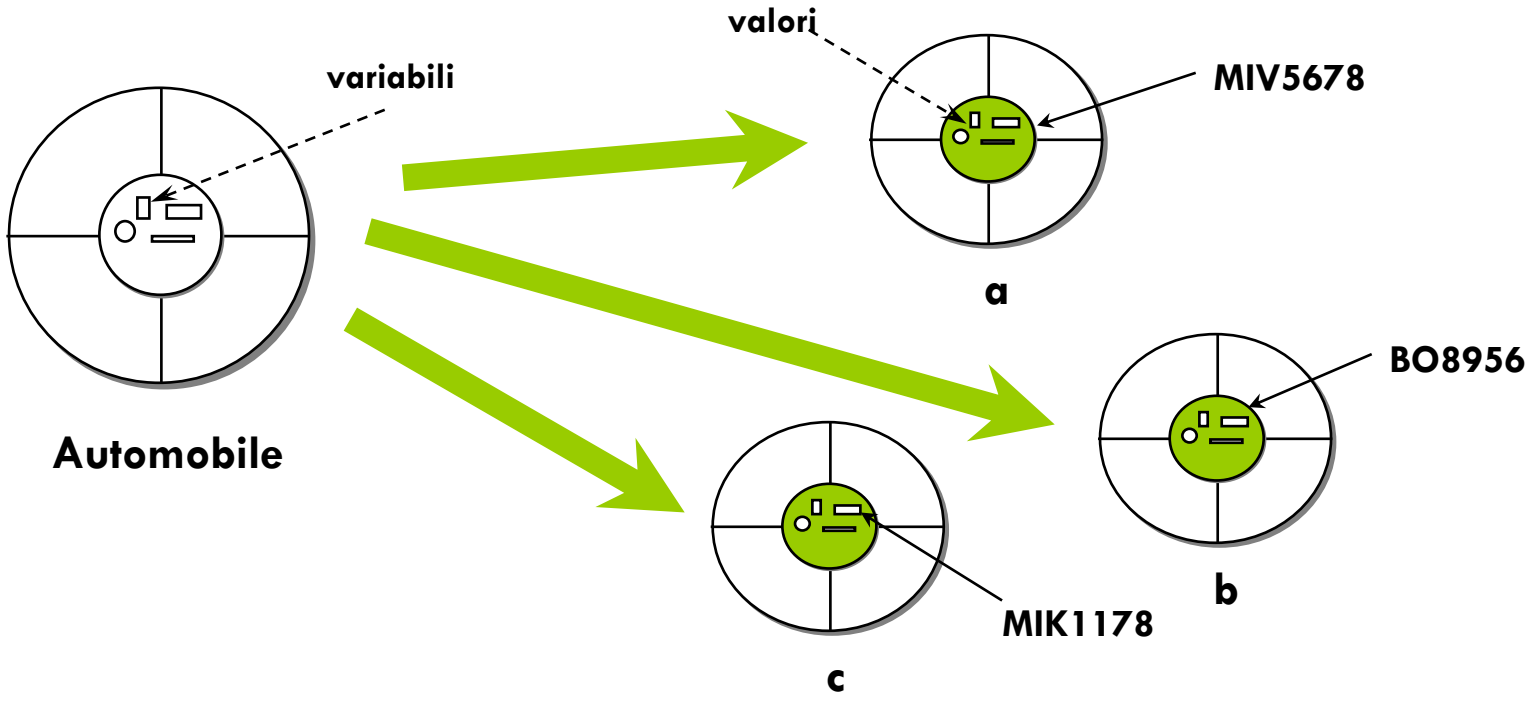
Esempio di definizione di un costruttore

```
Date today = Date(23,6,1983);
Date xmas(25,12,1990); // forma abbreviata
Date my_birthday; //errore : manca l'inizializzatoremanca
il terzo argomento

// Inizializzazione consigliata:
Date today = Date {23,6,1983};
Date xmas {25,12,1990}; // forma abbreviata
Date release1_0 {10,12}; //errore : manca
l'inizializzatoremanca il terzo argomento
```

Esempi di utilizzo del costruttore

Oggetto

- Un oggetto è un'istanza (esemplare) di una classe



Esemplari con identico comportamento e stato diverso

Overloading di costruttori

```
class Date {
    int d, m, y;

public:
    // ...
    Date(int, int, int); // giorno, mese, anno
    Date(int, int); // giorno, mese, anno attuale
    Date(int); // giorno, mese e anno attuali
    Date(); //default: oggi
    Date(const char*); // data come stringa
};
```

Sovraccarimento di costruttori

```
Date today {4}; // 4, mese e anno attuali
Date july4 {"July 4, 1983"};
Date guy {5,11}; // 5, Novembre, oggi
Date now; // defalut: oggi
Date start {}; // default: oggi
```

Chiamata di costruttori

Esempio di implementazione

```
class Date {
   int d, m, y;

public:
    Date(int dd, int mm, int yy) {
       d = dd;      m = mm;      y = yy;
     }

    void show_Date()    {
     cout << "Giorno " << d << " Mese " << m << " Anno " << y;
   }

};
```

Esempio di implementazione interna di metodi e costruttori

Esempio di implementazione

```
class Date {
   int d, m, y;

public:
    Date(int dd, int mm, int yy);
    void show_Date();

};


Date::Date(int dd, int mm, int yy)
{
  d = dd;
  m = mm;
  y = yy;
}

void Date::show_Date()
{
  cout << "Giorno " << d << " Mese " << m << " Anno " << y;
}
```

uso di scope ::

Esempio di implementazione esterna di metodi e costruttori

Esercizio

- Scrivere una classe `Punto2D`
  - campi
    - variabili del tipo `double`
  - costruttori
    - default
    - due punti iniziali
  - metodi
    - `get_punto2D`
      - valore attuale del punto
    - `translate_punto2D`
      - trasla il punto di una quantità definita dall'untente
    - `distanza_origine`
      - calcola la distanza euclidea dall'origine