

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Tipi definiti dall'utente: la classe Date Introduzione

Stefano Ghidoni



DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE



Agenda

- Progettare una classe con un esempio: la classe date
- Helper function
- Funzioni membro



Progettazione di una classe

- Ora creiamo la classe Date per rappresentare e manipolare le date
- Processo iterativo
 - A ogni iterazione rendiamo la classe più utile e sicura
 - Simile al reale processo di sviluppo del software
 - Ogni iterazione prevede una fase di test!



La classe Date

- Quali variabili membro saranno necessarie?
 - Di che tipo?

```
int giorno
int mese
int anno
```



La classe Date - 1

```
struct Date
{
    int y;          // anno
    int m;          // mese
    int d;          // giorno
};
```

```
Date today;
```

Nomi troppo corti?
Perché?

Non sono nomi molto evocativi, ma lo scope è piccolo, quindi sono accettabili

```
today.y = 2005;
today.m = 12;
today.d = 24;
```

Date:

y:	2005
m:	12
d:	24



- Cosa possiamo fare con oggetti di questa struct?
 - Tutto!
- L'utilizzo è lento e non molto sicuro

```
Date x;  
  
x.y = -3;  
x.m = 13;  
x.d = 32;
```

// questo codice è sintatticamente corretto? Compila?
Esegue?



```
Date y;  
  
y.y = 2000;  
y.m = 2;  
y.d = 29;
```

// questo codice è sintatticamente corretto? Compila?
Esegue?



```
Date y;  
  
y.y = 2000;  
y.m = 2;  
y.d = 29;
```

// questo codice è sintatticamente corretto? Compila?
Esegue?

// L'anno 2000 è bisestile?



Helper function

- Helper function: funzioni che eseguono le operazioni più comuni
 - Al posto nostro!
 - Programmate una sola volta
 - **Debuggate una sola volta!**
- Quali operazioni potremmo implementare?
 - Inizializzazione!
 - Incremento di un giorno

Da un punto di vista sintattico, una helper function è una funzione esterna, che però aiuta a gestire gli oggetti (ad esempio di classe Date).

Spesso c'è un argomento di una certa classe.



Helper function

```
void init_day(Date& dd, int y, int m, int d)
{
    // verifica che y, m, d siano una data corretta
    // se sì, inizializza
}

void add_day(Date& dd, int n)
{
    // incrementa dd di n giorni
}
```



Uso delle helper function

```
void f()
{
    Date today;           Today non inizializzato e usato
    // ...
    cout << today << '\n'; // << definito per Date
    // ...
    init_day(today, 2008, 3, 30);
    // ...
    Date tomorrow;
    tomorrow.y = today.y;   Tomorrow costruito a mano
    tomorrow.m = today.m;
    tomorrow.d = today.d + 1; 32/10/2025, lol
    cout << tomorrow << '\n';
}
```

Codice molto scorretto, anche se a prima vista non sembra (ancora peggio!)



Helper function?

- Esempio precedente:
 - Le helper function sono a disposizione
 - Possono essere bypassate
- È necessario rendere più visibili le helper function
- Sarebbe una buona idea rendere il loro uso obbligatorio

Ponendo le funzioni all'interno della classe si rendono più visibili e obbligatorie



Date – 2 (con funzioni membro)

```
struct Date
{
    int y, m, d;
    Date(int y, int m, int d);
    void add_day(int n);
};
```

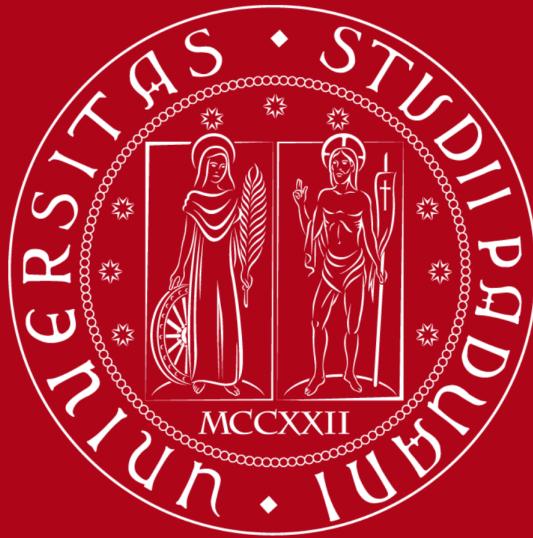
- **Funzioni membro**

- Sono parte integrante della classe

- Scope **di classe** → non di oggetto, quindi può accedere a membri privati di oggetti non `*this*`

Reminder:

Un costruttore è una funzione speciale con lo stesso nome della classe, ha dei parametri e nessun output esplicito, perché restituisce un oggetto della classe stessa.



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Tipi definiti dall'utente: la classe Date Introduzione

Stefano Ghidoni



DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE