



# C++



## Funzioni avanzate del linguaggio

Giulio Angiani  
I.I.S. "Blaise Pascal" - Reggio Emilia



# Overloading di operatori



# Cosa significa "overloading di operatori" ?

il cosiddetto **operator overloading** è una tecnica di sovrascrittura dei comportamenti degli operatori standard del linguaggio

Viene applicato laddove è necessario specificare al compilatore dei comportamenti **non noti** o per **modificare** comportamenti noti.

per esempio

- somma di due tipi di dato definiti dall'utente
- ridefinizione del concetto di uguaglianza fra elementi



# Somma di numeri interi

- il tipo **int** è predefinito in C++
- il concetto di **somma** è noto al compilatore (e anche a noi...)

```
int a = 3;  
int b = 5;  
int c = a+b;  
cout << "Somma = " << c << endl;
```

C++

Somma = 8

OUTPUT



# Somma di stringhe

- il tipo **string** è predefinito in C++
- il concetto di **somma** è noto al compilatore come "concatenazione"

```
string x = "Ciao";  
string y = "Ragazzi!";  
string z = x + " " + y;  
cout << z << endl;
```

C++

Ciao Ragazzi!

OUTPUT



## title: Differenza di stringhe

- il concetto di **differenza** di stringhe però NON è definito

```
string x = "Ciao";  
string y = "Ragazzi!";  
string z = x - y;
```

C++

```
esempio0.cpp:18:8: error: no match for 'operator-'  
(operand types are 'std::__cxx11::string {aka std::__cxx11::basic_string<char>}'  
and  
'std::__cxx11::string {aka std::__cxx11::basic_string<char>}')
```

COMPILER



# Differenza di stringhe

- Ridefiniamo il concetto di differenza fra stringhe

```
string operator-(string x, string y) {  
    return "Uhm! che operazione bizzarra!!!";  
}
```

C++

```
string x = "Ciao";  
string y = "Ragazzi!";  
string z = x - y;  
cout << z << endl;
```

```
Uhm! che operazione bizzarra!!!
```

OUTPUT



# Overloading su TDA

definiamo il tipo dato PuntoCartesiano

```
struct PuntoCartesiano {  
    float x;  
    float y;  
}  
  
PuntoCartesiano p1 {3, 4};  
PuntoCartesiano p2 {1, 8};  
  
PuntoCartesiano p3 = p1 + p2;
```

C++

```
esempio0.cpp:35:26: error: no match for 'operator+'  
(operand types are 'PuntoCartesiano' and 'PuntoCartesiano')
```

COMPILER





# Overloading su TDA

ridefiniamo il concetto di somma fra punti

qual è la somma di due punti?

in senso scalare?

in senso vettoriale?

```
// somma in senso scalare  
PuntoCartesiano operator+(PuntoCartesiano a, PuntoCartesiano b) {  
    PuntoCartesiano result = {a.x + b.x, a.y + b.y};  
    return result;  
}
```

C++

per esercizio al lettore lasciamo implementare la somma vettoriale



# Overloading su TDA

a questo punto possiamo sommare due numeri e stampare il risultato...

ma se sarebbe meglio poterlo stampare con una sola operazione....

```
PuntoCartesiano p3 = p1 + p2;  
cout << p3.x << ":" << p3.y << endl;
```

4:12



# Overloading su TDA

ridefiniamo anche la stampa

```
// ridefinisco l'operatore <<
ostream& operator<<(ostream& out, PuntoCartesiano p) {
    out << p.x << ":" << p.y;
    return out;
}
```

C++

la funzione prende in ingresso uno **stream** e l'oggetto da stampare e restituisce lo stesso stream con "intubata" la rappresentazione dell'oggetto.

4:12

OUTPUT





Giulio Angiani  
I.I.S. "Blaise Pascal" - Reggio Emilia