

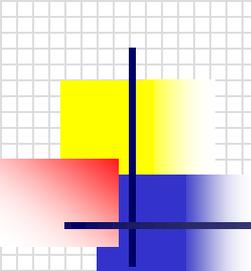
# Istituto Tecnico Industriale "M. M. Milano" Polistena

Classe III D a.s. 2015/2016



# C++

*Guida Base*



# Le Origini

---

- ✚ Il Linguaggio C++ è il più importante linguaggio di programmazione moderno per la produzione di software commerciale
- ✚ Fu Dennis Ritchie intorno agli anni sessanta a realizzare il primo prototipo di compilatore per il linguaggio C
- ✚ Nei primi anni ottanta, Bjarne Stroustrup arricchì il linguaggio C con nuove funzionalità, in grado di supportare la cosiddetta *programmazione orientata agli oggetti* (OOP)
- ✚ Dapprima chiamato C con classi, prese poi il nome di C++, per ricordare attraverso l'operatore classico di incremento del C, la natura evolutiva della sua realizzazione

# *Struttura di un programma*

Un programma in C++ è formato da una collezione di funzioni, tra cui ne deve esistere sempre una, chiamata *main*, che contiene le istruzioni da eseguire per prime.

*Specificazione dei file di intestazione (librerie)*

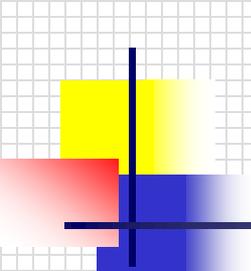
*main()*

{

*dichiarazione dei tipi e delle variabili*

*istruzioni*

}



# *I file di inclusione*

I file di inclusione inseriscono nel codice sorgente il contenuto del file indicato mettendo a disposizione del programmatore le funzionalità di quella libreria.

In C++ i file di inclusione si dichiarano:

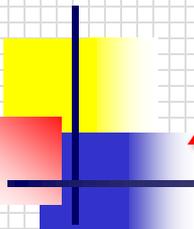
```
#include <nomefile.h>
```

Esempio:

```
#include <iostream.h>
```

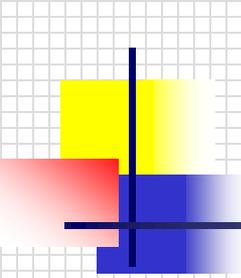
```
#include <conio.h>
```

Consentono di richiamare i file di libreria per la gestione dell'input/output attraverso il video e la tastiera



# Alcuni file di inclusione

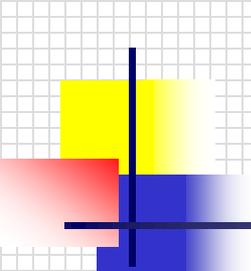
<code>conio.h</code>	Contiene diverse funzioni per la gestione del video e della tastiera: <code>clrscr()</code> ; pulisce il video <code>gotoxy(x, y)</code> ; sposta il cursore nella posizione specificata dalle coordinate;
<code>iostream.h</code>	È indispensabile per gestire attraverso le classi <code>cin</code> e <code>cout</code> l'input e l'output da console;
<code>math.h</code>	Contiene la dichiarazione delle principali funzioni matematiche <code>sqrt(x)</code> ; restituisce la radice quadrata di x
<code>stdlib.h</code>	Contiene la dichiarazione di molte routine di uso comune: <code>random(n)</code> ; restituisce un valore intero casuale compreso tra 0 e n-1 <code>randomize()</code> ; inizializza il generatore di numeri casuali.



# using namespace std;

Un programma C++ molto complesso può essere costituito da più file ciascuno dei quali verrà compilato separatamente. I risultati della compilazione vengono assemblati da un programma che si chiama LINKER.

Se un identificatore (nome di variabile, di funzione eccetera) viene definito extern in un file F1 esso verrà riconosciuto anche da tutti gli altri file linkati assieme a F1. Si possono così creare dei conflitti se lo stesso identificatore viene ridefinito per errore in file diversi. Per evitare tali conflitti si consente di ridefinire lo stesso identificatore su file diversi a condizione di creare degli SPAZI DEI NOMI (namespace) e di porre gli identificatori su namespace diversi.



# *Variabili*

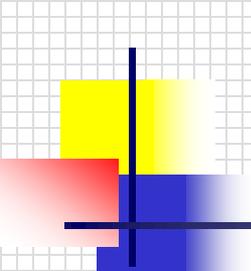
---

In C++ prima di usare un identificatore (variabile o costante) occorre effettuare la dichiarazione.

Una variabile è caratterizzata da:

- + un **nome simbolico**, che individua l'indirizzo dell'area di memoria che è stata allocata la variabile;
- + Un **tipo**, che determina la quantità di memoria che il compilatore associa alla variabile;
- + Il **valore** che assume la variabile in un certo istante;

In C++ l'identificatore sente la differenza tra lettere maiuscole e minuscole (case-sensitive), per cui, per esempio, le variabili **Somma**, **SOMMA** e **somma** vengono considerate diverse



# ***I commenti nel programma***

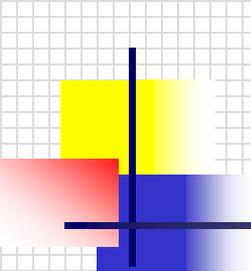
In C++ è possibile inserire dei commenti in qualsiasi punto del programma,

La combinazione `/*` viene impiegata per commenti lunghi che occupano solitamente più righe, tutti i commenti iniziano con `/*` e terminano con `*/`

Es.: `/* questo è un commento */`

La combinazione `//` permette di inserire una sola riga di commento, viene impiegata per brevi descrizioni e può essere utilizzata anche in una riga di istruzioni, questo tipo di commento termina con la fine della riga in cui è stato immesso.

Es.: `// anche questo è un commento`



# *Dichiarazione di variabili*

Per associare un tipo ad una variabile si utilizza la forma sintattica:

**tipo identificatore;**

Ad esempio con la dichiarazione:

**int anni;**

si indica al compilatore che per la variabile **anni** deve essere riservata una zona di memoria di 2 byte in un sistema a 16 bit, 4 byte in un sistema a 32 o 64 bit;

# *Dichiarazione di variabili*

## Alcuni tipi interi :

<b>int</b>	32 bit, valori da $-2.147.438.648$ a $+ 2.147.438.648$
<b>unsigned int</b>	32 bit, valori da 0 a 4.294.967.295

La dimensione e gli intervalli di rappresentazione dei tipi di variabili possono cambiare in base al compilatore e al computer utilizzati. I tipi di dati interi descritti prima si riferiscono ad un computer a 32 bit, se si fa riferimento a un computer a 16 bit la dimensione non è 32 bit bensì 16 bit.

# *Dichiarazione di variabili*

## Alcuni tipi reali :

<b>float</b>	32 bit, valori positivi e negativi con modulo compreso da $1,2E-38$ a $3,48E+38$
<b>double</b>	64 bit, valori positivi e negativi con modulo compreso da $2,2E-308$ a $1,8E+308$

**Le caratteristiche dei tipi di dat in C++  
possono cambiare in base al compilatore e  
al computer utilizzati**

# Operatori Booleani

Il Linguaggio C++ prevede gli operatori logici:

Simbolo	Operatore	Esempio	significato
!	negazione	!a	Non a
	disgiunzione	a  b	a o b
&&	congiunzione	a&&b	A e b

Questi tre operatori restituiscono 0 (zero) per falso e 1 per vero.

# Operatori algebrici

SEGNO	OPERAZIONE
+	Addizione
-	Sottrazione
*	Moltiplicazione
/	Divisione
%	Resto della divisione tra interi

Si può assegnare un valore a una variabile attraverso l'**operatore di assegnazione** (=), anche tramite il risultato di un'operazione algebrica;

Qualche esempio:

altezza = 5;

a = b/2;

risultato = a\*x\*x + b\*x + c;

# *Istruzioni di input/output*

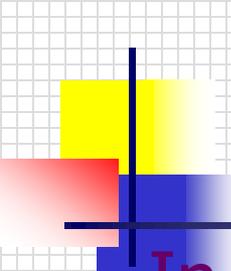
Dopo aver richiamato il file di intestazione `<iostream.h>` è possibile gestire l'input/output da tastiera e video utilizzando gli operatori `>>` e `<<` nonché le classi standard `cin` e `cout`

Codice	Significato
<code>cin&gt;&gt;anno;</code>	Permette di inserire da tastiera un valore nella variabile <code>anno</code>
<code>cout&lt;&lt;anno;</code>	Viene visualizzato il contenuto della variabile <code>anno</code>
<code>cout&lt;&lt;anno&lt;&lt;'\n';</code>	Viene visualizzato il contenuto della variabile <code>anno</code> e il cursore va alla linea successiva
<code>cout&lt;&lt;"questo messaggio";</code>	Viene visualizzato il contenuto della stringa racchiusa tra ""

# *Esempio di programma in C++*

/\* Inserito da tastiera il valore di un lato di un quadrato calcolare e poi visualizzare il perimetro e l'area. \*/

```
#include<iostream.h>
#include <conio.h>
main()
{
int lato, perim, area;
cout<<"inserisci il lato \n";
cin>>lato;
perim=lato*4;
area=lato*lato;
cout<<"il perimetro è " <<perim <<"\n";
cout<<"L'area è " <<area <<"\n";
cout<<"premi un tasto per uscire dall'esecuzione";
while (!kbhit());
}
/* le ultime due righe di codice sono utilizzate per far attendere i risultati su video,
per l'uso dell'istruzione kbhit è necessario includere il file conio.h */
```

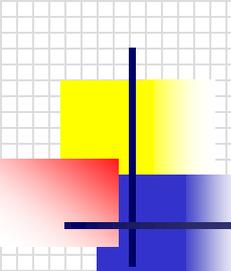


# ***Strutture di controllo***

In C++ le strutture di controllo si dividono in tre tipi:

- + Sequenza
- + Selezione
- + Iterazione

La **sequenza** è una struttura in cui le istruzioni che la compongono vengono eseguite una dopo l'altra nell'ordine indicato dove il blocco di istruzioni è racchiuso tra i simboli { e }



# ***Selezione (struttura condizionale)***

La selezione è una struttura che permette, al verificarsi o no di una condizione, di scegliere l'esecuzione delle istruzioni contenute in uno tra due blocchi distinti (eventualmente vuoti)

La rappresentazione formale è:

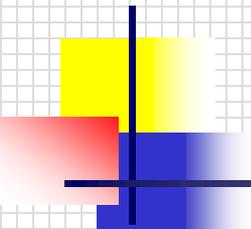
```
If (condizione)
    istruzione1;
else
    istruzione2;
```

Ovviamente la parte relativa all'else può anche mancare. Nel caso si debbano eseguire più istruzioni quando la condizione è vera o falsa, bisogna racchiuderle tra { e }

# *Esempio di programma con if..*

Inseriti due numeri interi, calcolare la differenza, il più grande meno il più piccolo.

```
#include<iostream.h>
#include <conio.h>
main()
{
int numa, numb, differenza;
cout<<"inserisci il primo numero \n";
cin>>numa;
cout<<"inserisci il secondo numero \n";
cin>>numb;
if (numa>numb)
differenza= numa - numb;
else
differenza= numb - numa;
cout<<"La differenza tra il più grande e il più piccolo e` " <<differenza <<"\n";
cout<<"premi un tasto per uscire dall'esecuzione";
while (!kbhit());
}
```



# C++

## The end

