



Introduzione al C

Informatica B

Il nostro primo programma in C

```
/* Questo è il nostro primo  
   programma in C */
```

```
#include <stdio.h>
```

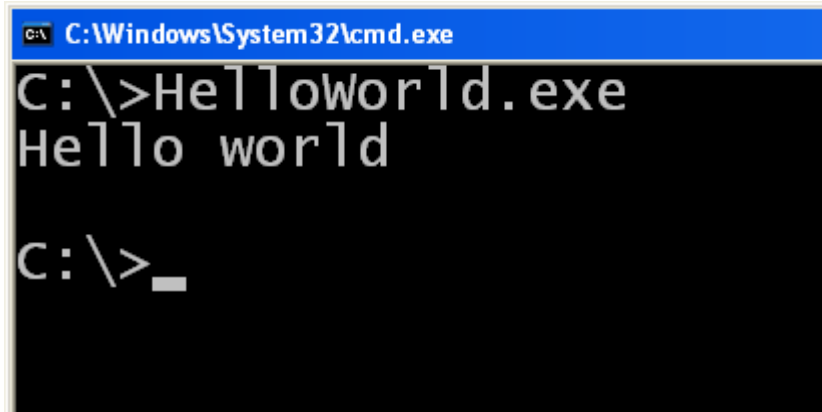
```
int main( )
```

```
{
```

```
    printf("Hello world\n");
```

```
    return 0;
```

```
}
```



A screenshot of a Windows command prompt window. The title bar reads "C:\Windows\System32\cmd.exe". The command prompt shows the command "C:\>HelloWorld.exe" being entered and executed. The output is "Hello world". The prompt then shows "C:\>_" indicating the command has finished and the prompt is ready for the next command.

Il nostro primo programma in C

```
/* Questo è il nostro primo  
programma in C */
```

```
#include <stdio.h>
```

```
int main()  
{  
    printf("Hello world\n");  
    return 0;  
}
```

- ❑ Questo è un **commento**
- ❑ Tutto ciò che viene racchiuso fra */** e **/* viene ignorato dal compilatore:

```
/* ... commento ... */
```

- ❑ */** **apre** il commento
- ❑ **/* **chiude** il commento
- ❑ Viene utilizzato per aggiungere note e/o descrivere parti del programma

Il nostro primo programma in C

/ Questo è il nostro primo programma in C */*

```
#include <stdio.h>
```

```
int main()
{
    printf("Hello world\n");
    return 0;
}
```

- ❑ Indica al compilatore che nel nostro programma verranno utilizzate delle **funzioni esterne**
- ❑ In questo caso specifica che queste funzioni si trovano nella **libreria `stdio.h`** (standard input output)
- ❑ Il C mette a disposizione del programmatore diverse librerie, cioè diverse collezioni di funzioni (I/O, operazioni matematiche, informazioni su data e ora, ecc.)
- ❑ Per usare una libreria nel nostro programma occorre specificarlo all'inizio:

```
#include <nome libreria>
```

Il nostro primo programma in C

```
/* Questo è il nostro primo  
programma in C */
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hello world\n");
```

```
    return 0;
```

```
}
```

- ❑ Ogni programma in C deve contenere una funzione principale, chiamata **main**
- ❑ Questa funzione contiene le istruzioni che verranno eseguite non appena il nostro programma viene caricato in memoria

Il nostro primo programma in C

```
/* Questo è il nostro primo  
programma in C */
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hello world\n");
```

```
    return 0;
```

```
}
```

- ❑ La sequenza di istruzioni della funzione main (detto anche **corpo**) deve essere racchiuso all'interno di **parentesi graffe**
- ❑ Ciascuna istruzione (**statement**) deve essere seguita da un **punto e virgola**
- ❑ Più in generale, in C una sequenza di istruzioni racchiuse da parentesi graffe costituisce una sorta di "macroistruzione" detta **blocco**:

```
{
```

```
    istruzione1;
```

```
    ...
```

```
    istruzionen;
```

```
}
```

Il nostro primo programma in C

```
/* Questo è il nostro primo  
programma in C */
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
printf("Hello world\n");
```

```
return 0;
```

```
}
```

- ❑ Questa istruzione stampa a video il messaggio "Hello world"
- ❑ Utilizza la funzione **printf** fornita dalla libreria **stdio.h**
- ❑ La sintassi della funzione è la seguente:

```
printf ( "messaggio" )
```

- ❑ Il messaggio da stampare deve essere racchiuso da ""
- ❑ **\n** è una sequenza speciale (**newline**) che può essere usato per mandare il cursore a capo

Il nostro primo programma in C

```
/* Questo è il nostro primo  
programma in C */
```

```
#include <stdio.h>
```

```
int main()  
{  
    printf("Hello world\n");  
    return 0;  
}
```

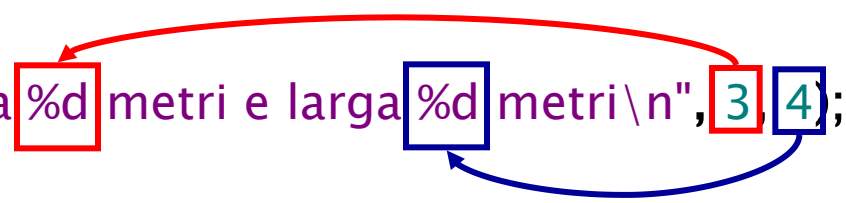
- ❑ L'istruzione **return** serve a terminare la funzione **main** ritornando un valore che rappresenta l'esito dell'esecuzione
- ❑ Nel caso della funzione **main** viene ritornato convenzionalmente il valore **0**, per indicare al sistema operativo che il programma è stato eseguito con successo
- ❑ Valori diversi possono essere usati per segnalare eventuali errori avvenuti

Output di un programma

```
/* Programma Room1.c */  
#include <stdio.h>
```

```
int main()  
{  
    printf("La mia stanza e' lunga %d metri e larga %d metri\n", 3, 4);  
    return 0;  
}
```

```
C:\>Room1.exe  
La mia stanza e' lunga 3 metri e larga 4 metri  
C:\>_
```

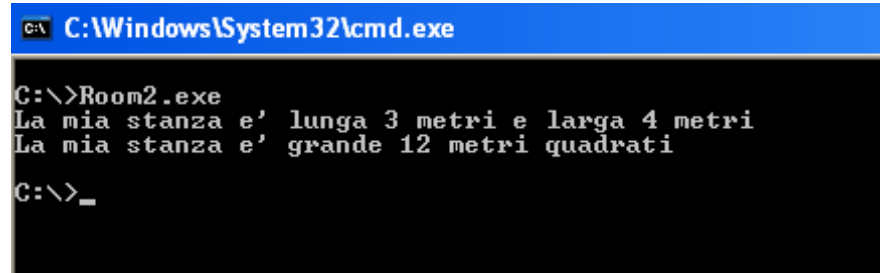


- ❑ La funzione `printf` consente di stampare anche dei valori numerici all'interno dei messaggi:
`printf("msg",valore1,valore2,...);`
- ❑ Il primo argomento, detto **stringa di controllo**, può contenere **caratteri di formato** (preceduti da `%`) che verranno associati ai restanti argomenti della `printf`
- ❑ `%d` indica che l'argomento da stampare è un **numero intero**

Semplici calcoli

```
/* Programma Room2.c */  
#include <stdio.h>
```

```
int main()  
{  
    printf("La mia stanza e' lunga %d metri e larga %d metri\n",3,4);  
    printf("La mia stanza e' grande %d metri quadrati\n",3*4);  
    return 0;  
}
```



```
C:\ Windows\System32\cmd.exe  
C:\>Room2.exe  
La mia stanza e' lunga 3 metri e larga 4 metri  
La mia stanza e' grande 12 metri quadrati  
C:\>_
```

- ❑ È anche possibile utilizzare come argomenti espressioni aritmetiche
- ❑ In questo caso, il calcolatore calcolerà l'espressione e poi utilizzerà il risultato come valore da stampare

Le variabili

- Affinché un programma sia in grado di risolvere una classe di problemi, occorre introdurre il concetto di **variabile**.

```
/* Programma Room3.c */
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int lung, larg, area;
```

```
    lung = 3;
```

```
    larg = 4;
```

```
    area = lung*larg;
```

```
    printf("La mia stanza e' lunga %d metri e larga %d metri\n",lung,larg);
```

```
    printf("La mia stanza e' grande %d metri quadrati\n",area);
```

```
    return 0;
```

```
}
```

Le variabili

```
/* Programma Room3.c */  
#include <stdio.h>
```

```
int main()  
{  
    int lung, larg, area;  
    lung = 3;  
    larg = 4;  
    area = lung*larg;  
    printf("La mia stanza e' lunga %d  
metri e larga %d metri\n",lung, larg);  
    printf("La mia stanza e' grande %d  
metri quadrati\n",area);  
    return 0;  
}
```

- ❑ Per usare una variabile in C, occorre **dichiararla**
- ❑ La **dichiarazione** di una variabile deve specificarne il **tipo** e il nome:

tipo nome1, nome2, ...;

- ❑ **int** indica al compilatore che la variabile conterrà un numero intero

Le variabili

```
/* Programma Room3.c */  
#include <stdio.h>
```

```
int main()  
{  
    int lung, larg, area;  
    lung = 3;  
    larg = 4;  
    area = lung*larg;  
    printf("La mia stanza e' lunga %d  
metri e larga %d metri\n",lung,larg);  
    printf("La mia stanza e' grande %d  
metri quadrati\n",area);  
    return 0;  
}
```

- ❑ In C è possibile assegnare un valore ad una variabile con la sintassi:

nome = <espressione>

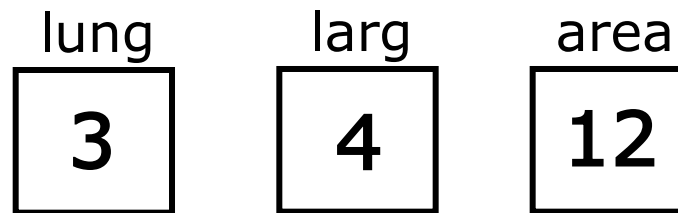
- ▶ Nome è il nome di una variabile
- ▶ Espressione è una qualsiasi espressione aritmetica valida in C (può contenere variabili)

- ❑ Il risultato dell'espressione viene messo nella variabile

nome ← <espressione>

Le variabili

```
int lung, larg, area;  
lung = 3;  
larg = 4;  
area = lung*larg;
```



Lettura dati dal terminale

- ❑ Come posso fornire informazioni al mio programma in C?

```
/* Programma Room4.c */
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int lung, larg, area;
```

```
    scanf("%d",&lung);
```

```
    scanf("%d",&larg);
```

```
    area = lung*larg;
```

```
    printf("La mia stanza e' lunga %d metri e larga %d metri\n",lung,larg);
```

```
    printf("La mia stanza e' grande %d metri quadrati\n",area);
```

```
    return 0;
```

```
}
```

Letture dati dal terminale

```
/* Programma Room4.c */  
#include <stdio.h>
```

```
int main()  
{  
    int lung, larg, area;
```

```
    scanf("%d",&lung);  
    scanf("%d",&larg);
```

```
    ...
```

- ❑ La lettura di informazioni dal terminale avviene tramite la funzione **scanf** fornita dalla libreria **stdio.h**
- ❑ La sintassi da utilizzare è:

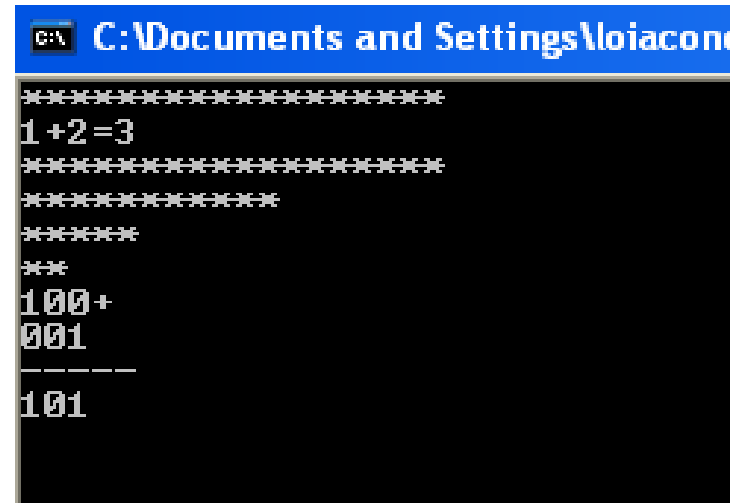
scanf("str ctrl", &variabile)
- ❑ Il primo parametro è la **stringa di controllo**
- ❑ È possibile definire un **pattern** input (es. leggere più variabili, leggere variabili in una specifica sequenza)
- ❑ In generale è preferibile utilizzare stringhe di controllo semplici e leggere una sola variabile per volta

Ancora su printf

...

```
printf("*****");  
printf("\n1+2=%d\n",1+2);  
printf("*****\n");  
printf("*****\n");  
printf("*****\n");  
printf("**\n");  
printf("%d%d%d+\n",1,0,0);  
printf("%d%d%d\n",0,0,1);  
printf("-----\n%d%d%d\n",1,0,1);
```

...



```
C:\Documents and Settings\loiacono>  
*****  
1+2=3  
*****  
*****  
*****  
**  
100+  
001  
-----  
101
```

Struttura di un programma C

