

Programmazione in C

Dichiarazioni

```
<tipo> <nome_var>;  
ES int pippo;  
char a;
```

Inizializzazione

```
<VAR> = <espressione>;  
ES a = 'b';  
pippo = 10;  
double pi = 3.14;
```

Input/Output

Output:

STRINGA COSTANTE

```
printf("Ciao!");
```

STAMPA VARIABLE

```
printf("%d", pippo);
```

- specificatore di formato

Input

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

- indirizzo come parametro

Casting

```
VAR = (tipo) espressione;
```

```
ES a = (char) pippo;
```

Enum

```
enum <nome> { <v1>, ..., <vn> };
```

ES

```
enum colore { bianco, rosso, verde };
```

```
enum colore bandiera;
```

Strutture di Controllo

Selezione

```
if (<condizione>) { }
```

```
else { }
```

if-else

```
if (<cond>) { }
```

```
else if (<cond2>) { }
```

```
...
```

switch-case

```
switch (<var>)
```

```
{
```

```
case 1:
```

```
    :  
    break;
```

```
default:
```

```
}
```

While

```
while (<cond>)  
{ }
```

do-while

```
do {  
    } while (<cond>);
```

for

```
for (init; cond; incremento)  
{ }
```

```
ES for (int i = 0; i < 10; i++)
```

Sottoprogrammi

PROCEDURA

```
void <nome> (<param>...);
```

FUNZIONE

```
<tipo> <nome> (<param>...);
```

ARRAY

```
tipo nome_array [dim];
```

LETTURA

```
VAL = v [INDICE];
```

SCRITTURA

```
v [INDICE] = VAL;
```

MATRICI

```
tipo nome_array [dimR] [dimC];
```

LETTURA

```
VAL = v [i-RIGHE] [i-COLONNE];
```

SCRITTURA

```
v [i-RIGHE] [i-COLONNE] = VAL;
```

STRUTTURE

```
struct nome {
```

```
    tipo_campo1 nome;
```

```
    tipo_campo2 nome;
```

```
    ...  
    tipo_campoN nome;
```

```
};
```

VARIABILI STRUTTURA

```
struct nome_struct nome_var;
```

ACCESSO AI CAMPI

```
VAR. <nome_campo>;
```

DEFINIZIONE TIPO

```
typedef struct <nome_struct>  
    <nome_tipo>;
```

Stringhe

DEFINIZIONE

```
char nomeStr [dim];
```

FUNZIONI string.h

```
strlen (<str>);
```

```
strcpy (<dest>, <src>);
```

```
strcmp (<str>, <str>);
```

Puntatori

DICHIARAZIONE

```
nome_tipo * nome;
```

ES

```
int * a;
```

```
a = &pippo;
```

ALLOCAZIONE DINAMICA

```
Void* malloc (<dimensioni>);
```

ES

```
int * p = (int*) malloc (sizeof(int));
```