

Linguaggio C

strutture di controllo:
strutture sequenziali e condizionali

Università degli Studi di Brescia

Docente: Massimiliano Giacomini

Strutture di controllo

- Controllano il flusso di esecuzione delle istruzioni: quali istruzioni devono essere eseguite sulla base dello stato di esecuzione
- Tipologie di strutture di controllo:
 - struttura sequenziale
 - istruzioni di selezione (condizionali)
 - istruzioni cicliche (iterative)

Struttura sequenziale

- Ciascuna istruzione è separata da punto e virgola
- Se non vi sono istruzioni condizionali o iterative, le istruzioni sono eseguite in sequenza
- Un blocco di istruzioni può essere racchiuso tra parentesi graffe senza punto e virgola: in questo caso il blocco viene trattato come una sorta di “istruzione singola” composta dalle istruzioni componenti (vedi i prossimi lucidi)

```
...  
{ int primo, secondo, somma;  
  printf("Inserisci due numeri interi\n");  
  scanf("%d%d",&primo,&secondo);  
  somma=primo+secondo;  
  printf("Somma uguale a %d\n",somma);  
}
```

Strutture condizionali

- Permettono di eseguire un'istruzione (o un blocco di istruzioni) solo se si verifica una data condizione
- La condizione corrisponde ad un valore booleano (vero o falso)
- In C, esistono vari tipi di strutture condizionali...

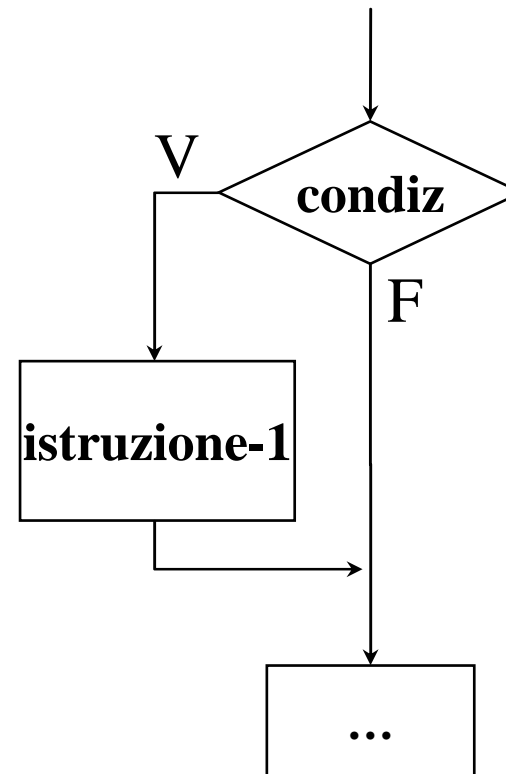
Struttura condizionale semplice: if

Sintassi

```
if (condiz)
    istruzione-1;
...
```

- *condiz* è una qualunque espressione aritmetica, ma tipicamente è costruita con operatori relazionali
p.es. $(a < b)$
- *istruzione-1* può essere anche un blocco istruzioni

Semantica

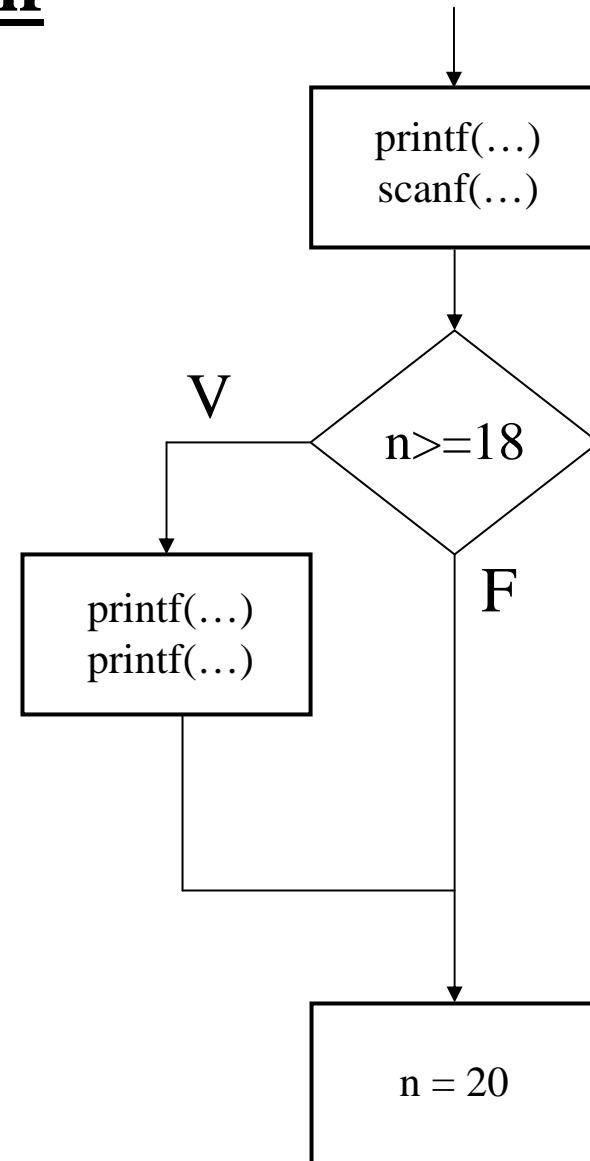


- se *condiz* ha valore diverso da zero (vera), esegue istruzione-1
- in ogni caso si passa poi alle istruzioni successive

Struttura condizionale semplice: if

Esempio

```
printf("Inserisci età ");  
scanf("%d", &n);  
  
if(n >= 18)  
{  
    printf("Hai %d anni ", n);  
    printf("quindi sei maggiorenne\n");  
}  
  
n=20;
```



Operatori relazionali su tipi numerici

- Servono per esprimere le condizioni che riguardano “numeri”

$==$	uguale	ES: $a==b$ vera se a uguale a b
$<$	minore	ES: $a<b$ vera se a minore di b
$<=$	minore o uguale	ES: $a<=b$ vera se a min. o ug. a b
$>$	maggiore	ES: $a>b$ vera se a maggiore di b
$>=$	maggiore o uguale	ES: $a>=b$ vera se a magg. o ug. a b
$!=$	non uguale	ES: $a!=b$ vera se a diverso da b

- Attenzione a non confondere:
 - l'operatore di assegnamento =
 - l'operatore di uguaglianza ==

Esercizio 1

Scrivere un frammento di codice C che acquisisce da tastiera un numero intero e stampa un messaggio di errore se il numero inserito è negativo.

Esercizio 1

Scrivere un frammento di codice C che acquisisce da tastiera un numero intero e stampa un messaggio di errore se il numero inserito è negativo.

```
int n;
```

```
...
```

```
printf("Inserire un numero intero positivo\n");
```

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

```
if(n < 0)
```

```
    printf("Dovevi inserire un numero positivo!\n");
```

Esercizio 2

Scrivere un frammento di codice C che, data una variabile intera n , la trasforma nel suo valore assoluto.

Esercizio 2

Scrivere un frammento di codice C che, data una variabile intera n , la trasforma nel suo valore assoluto.

```
int n;
```

```
...
```

```
if(n < 0)
```

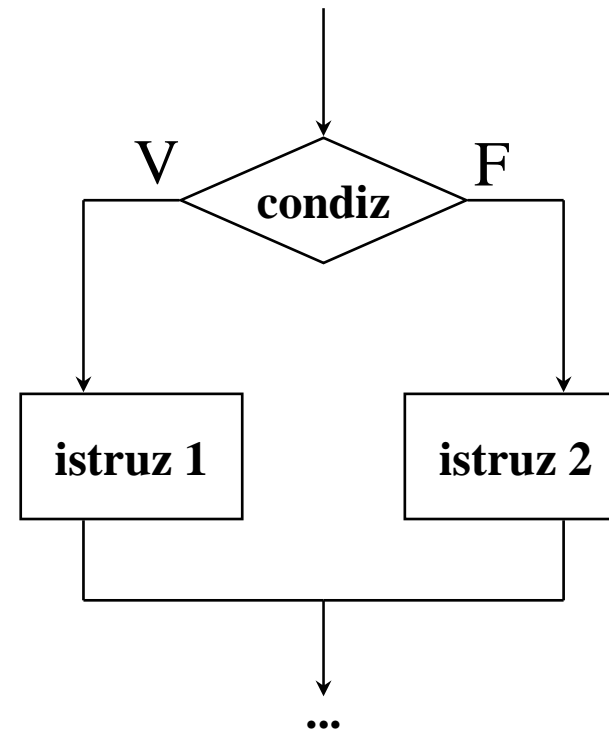
```
    n = -n;
```

Strutture condizionali: if-else

Sintassi

```
if (condiz)
    istruzione-1;
else
    istruzione-2;
...
```

Semantica



- Se *condiz* è non nulla (vera) si esegue istruzione-1, altrimenti (se falsa) si esegue istruzione-2; successivamente l'esecuzione prosegue con le istruzioni successive
- Anche in questo caso le istruzioni 1 e 2 possono essere blocchi

Strutture condizionali: if-else

Esempio

```
printf("la variabile a contiene un numero ");
```

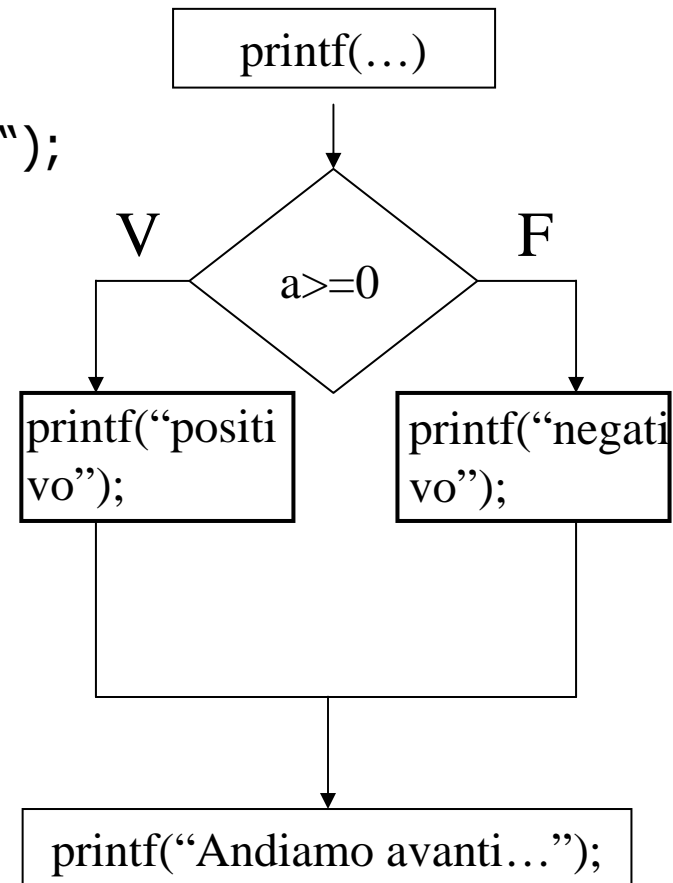
```
if(a >= 0)
```

```
    printf("positivo\n");
```

```
else
```

```
    printf("negativo\n");
```

```
printf("Andiamo avanti...");
```



Condizioni IF-ELSE nidificate

- Nel caso di istruzioni IF-ELSE “una dentro l’altra”, un *else* si riferisce sempre all’*if* più vicino (privo di else)

Esempi

```
if(x>y)
    if(x>0)
        printf("il maggiore è x ed è positivo\n");
    else printf("il maggiore è x ed è negativo\n");
```

```
if(x>y){
    if(x>0)
        printf("il maggiore è x ed è positivo\n");
}
else printf("il maggiore è y\n");
```

Errori comuni nell'uso di IF-ELSE

- Confondere = (operatore di assegnamento)
con == (operatore relazionale di uguaglianza)

Esempio 1:

```
if(a=1)
```

```
    printf("a=1\n");
```

stampa in ogni caso “a=1” e assegna alla variabile *a* il valore 1

Esempio 2:

```
if(a=0)
```

```
    printf("a=0\n");
```

assegna alla variabile *a* il valore 0 e non esegue la stampa

Errori comuni nell'uso di IF-ELSE

- Mettere il punto e virgola subito dopo *if* (o *else*): in questo caso l'istruzione che segue *if* è l'istruzione vuota (seguita da ;)

Esempio:

```
if(a==1);  
    printf("a=1\n");
```

se `a==1` esegue l'istruzione vuota (cioè niente)!

Poi, in ogni caso stampa “a=1”!

Operatori logici

- Permettono di esprimere una condizione complessa, costruita a partire da condizioni più semplici
- Il “valore di verità” della condizione complessa dipende dai valori di verità delle condizioni più semplici

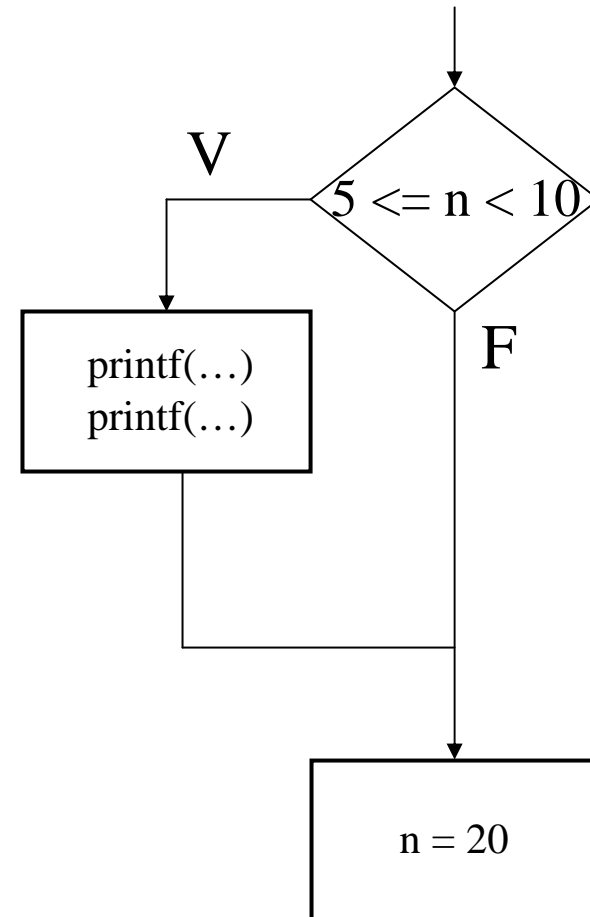
! NOT (operatore unario)
!cond vera se *cond* falsa

&& AND (operatore binario)
(c1 && c2) vera se *c1* e *c2* sono vere entrambe

|| OR (operatore binario)
(c1 || c2) vera se almeno una tra *cond1* e *cond2*
 è vera

Esempio

```
if(n >= 5 && n<10)
{
    printf("n è nell'intervallo ");
    printf("[5, 10)\n");
}
n=20;
```



Esercizio 3

Acquisire da tastiera un carattere. Determinare se si tratta di una vocale o di una consonante.

Esercizio 3

Acquisire da tastiera un carattere. Determinare se si tratta di una vocale o di una consonante.

```
char n;  
  
printf("Inserisci un carattere\n");  
  
scanf("%c", &n);  
  
if(n=='a' || n=='e' || n=='i' || n=='o' || n=='u')  
    printf("Il carattere è una vocale\n");  
else printf("Il carattere è una consonante\n");
```

Esercizio 4

Scrivere un programma che determini il massimo valore tra quattro numeri acquisiti da tastiera.

Esercizio 4

Scrivere un programma che determini il massimo valore tra quattro numeri acquisiti da tastiera.

L'IDEA...

- 1) Acquisisci i numeri (x, y, v, z)
- 2) Se $(x > y \ \&\& \ x > v \ \&\& \ x > z)$
 il maggiore è x
 altrimenti //siamo sicuri che x non è massimo
 se $(y > v \ \&\& \ y > z)$
 il maggiore è y
 altrimenti //siamo sicuri che né x né y siano massimi
 se $(v > z)$
 il maggiore è v
 altrimenti il maggiore è z

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

main(){
    int x,y,v,z, max;

    printf("Inserisci x\n");
    scanf("%d",&x);
    ... //lo stesso per y,v,z

    if(x>y && x>v && x>z)
        max=x;
    else if(y>v && y>z)
        max=y;
    else if(v>z)
        max=v;
    else max=z;

    printf("Massimo = %d\n", max);
    system("pause");
}
```

UN'ALTRA IDEA...

- 1) Acquisisci i numeri (x, y, v, z)
- 2) `max=x;`
 `se(y>max) max=y;`
 `se(v>max) max=v;`
 `se(z>max) max=z;`


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

main(){
    int x,y,v,z, max;

    printf("Inserisci x\n");
    scanf("%d",&x);
    ... //lo stesso per y,v,z

    max=x;
    if(y>max)
        max=y;
    if(v>max)
        max=v;
    if(z>max)
        max=z;

    printf("Massimo = %d\n", max);
    system("pause");
}
```

Esercizio 5

Scrivere un programma che determini se un anno acquisito da tastiera è bisestile. Un anno è bisestile se il suo numero è divisibile per 4, con l'eccezione che gli anni secolari (quelli divisibili per 100) sono bisestili solo se divisibili per 400

Esercizio 5

Scrivere un programma che determini se un anno acquisito da tastiera è bisestile. Un anno è bisestile se il suo numero è divisibile per 4, con l'eccezione che gli anni secolari (quelli divisibili per 100) sono bisestili solo se divisibili per 400

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

main(){
    int anno;
    printf("Inserisci l'anno\n");
    scanf("%d", &anno);
    if( ((anno % 100 !=0) && (anno % 4 ==0)) || anno%400==0)
        printf("Anno bisestile\n");
    else printf("Anno non bisestile\n");
    system("pause");
}
```

Strutture condizionali: il costrutto switch

```
switch (espressione)
{
    case <costante_1>: case <costante_2> ...: istruz_1_1;
                                                istruz_1_2;
                                                ...
    case <....>: istruz_2_1;
        ...
    ...
    default: ... ←———— opzionale
}
...
```

- *espressione* deve restituire un valore intero
- se espressione è pari a *costante_i*, vengono eseguite le istruzioni corrispondenti e tutte quelle dei **case** successivi (cfr. istruz. *break*)
- le istruzioni del caso *default* vengono eseguite “a prescindere”

Istruzione break:

- Causa l'uscita dall'istruzione switch: usata tipicamente come ultima istruzione tra quelle relative ad un caso per evitare l'esecuzione delle istruzioni dei *case* successivi

Esempio

```
switch(n){  
    case 0: printf("Valore nullo\n");  
            break;  
    case 1: printf("Valore 1\n");  
            break;  
    default: printf("Valore diverso da 0 e da 1\n");  
}
```

IF VS. SWITCH

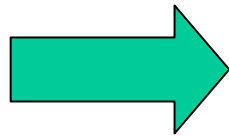
- Ovviamente, la funzione del costrutto switch può essere ottenuta utilizzando IF-THEN-ELSE
- Il costrutto switch può essere più conveniente quando la scelta dipende da quale specifico valore assume una variabile intera, ovvero:
 - se `val == 1` allora...
 - se `val == 2` allora...
 - se `val == 3` allora...
 - altrimenti...
- Notare che il costrutto switch è molto rigido per quel che riguarda le condizioni da verificare, ad esempio non è possibile confrontare due variabili
[p.es. `(a < b)` non può essere espressa come condizione case]

Esercizio 6

Sviluppare un programma che chieda all'utente di scegliere una lingua (fra italiano, inglese, ecc.) e produca a video la scritta “Ciao mondo” nella lingua prescelta.

Esercizio 6

Sviluppare un programma che chieda all'utente di scegliere una lingua (fra italiano, inglese, ecc.) e produca a video la scritta “Ciao mondo” nella lingua prescelta.



Trattandosi di un problema di “scelta multipla”, la soluzione più appropriata è utilizzare il costrutto switch


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

main(){
char scelta;

printf("Scegli la lingua: i per italiano, e per inglese, f per francese\n");
scanf("%c",&scelta);

switch(scelta){
    case 'i':case 'I': printf("Ciao mondo\n");
                    break;
    case 'e':case 'E': printf("Hello world\n");
                    break;
    case 'f':case 'F': printf("Bonjour le monde\n");
                    break;
    default: printf("Non conosco questa lingua!\n");
}

system("pause");
}
```

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

Proviamo a togliere qualche break...

```
main(){
char scelta;

printf("Scegli la lingua: i per italiano, e per inglese, f per francese\n");
scanf("%c",&scelta);

switch(scelta){
    case 'i':case 'I': printf("Ciao mondo\n");
                    //break;
    case 'e':case 'E': printf("Hello world\n");
                    break;
    case 'f':case 'F': printf("Bonjour le monde\n");
                    //break;
    default: printf("Non conosco questa lingua!\n");
}

system("pause");
}
```

Esercizi per casa

- 1) Acquisire da tastiera un carattere. Se il carattere è minuscolo stampare a video il corrispondente maiuscolo, se il carattere è maiuscolo stampare a video il corrispondente minuscolo.
Suggerimento: si determini preliminarmente il valore di 'a' - 'A' mediante un semplice programma C.
- 2) Modificare il programma che determina se un carattere inserito è una vocale oppure una consonante, in modo che funzioni per caratteri sia maiuscoli sia minuscoli.
- 3) Calcolare l'intervallo di dispersione tra quattro numeri acquisiti da tastiera (l'intervallo di dispersione è la differenza tra il massimo e il minimo numero).