PASCAL
LA SELEZIONE: IF

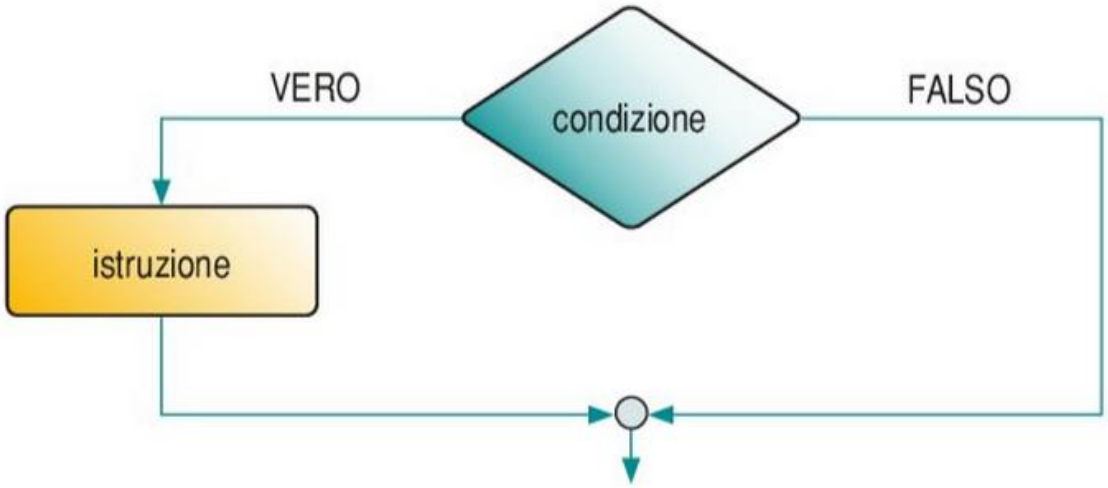
TRATTO DA CAMAGNI-NIKOLASSY, CORSO DI INFORMATICA, VOL. 1, HOEPLI

Informatica

La selezione semplice: *if..then*

La selezione semplice: *if..then*

- L'*istruzione di selezione semplice* permette l'esecuzione di alcune istruzioni solo se il valore di istruzione di test (o condizione) ha esito positivo (o VERO)
- In caso contrario (FALSO) non viene eseguita nessuna istruzione



- condizione VERA: si esegue il ramo sinistro e quindi l'istruzione
- condizione FALSA: si esegue il ramo destro, senza fare nulla.

If..then in Pascal

- La codifica in linguaggio Pascal è:

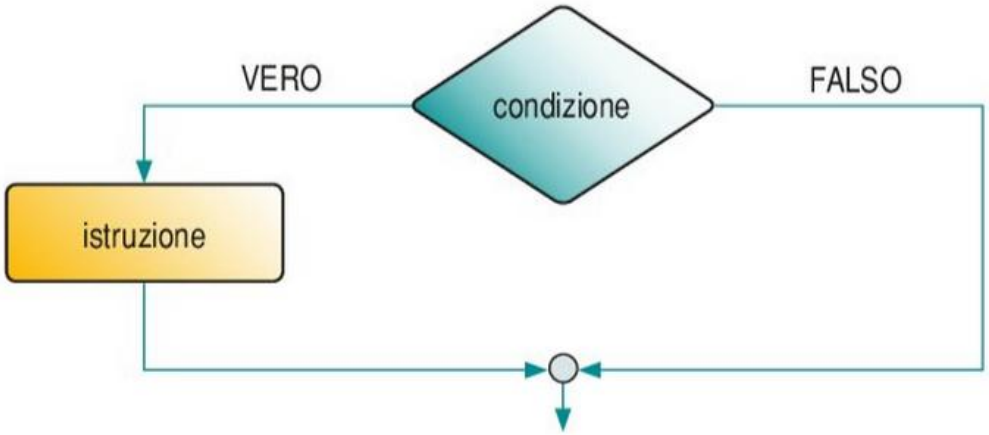
```
if(condizione)
  then istruzione;
```

If..then: Codifica vs Schema a blocchi

- La codifica in linguaggio Pascal è:

```
if(condizione)
  then istruzione;
```

vs



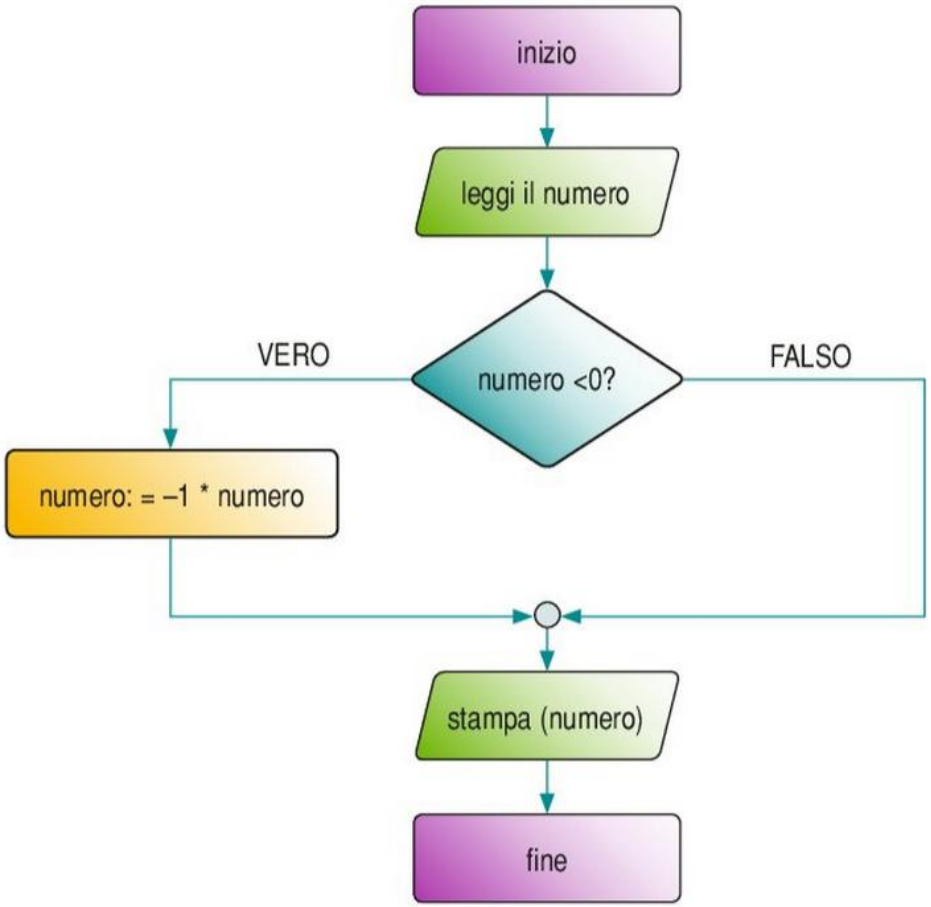
La condizione

Una condizione è un'espressione del linguaggio rappresentata da due elementi messi a confronto da un operatore relazionale.
Nella tabella sono riportati i principali operatori relazionali del linguaggio Pascal.

Operatore	Significato	Esempio
>	maggiore	var1 > var2
<	minore	var1< var2
>=	maggiore uguale	var1 >= var2
<=	minore uguale	var1 <= var2
=	uguale	var1 = var2
<>	diverso	var1<> var2

Esempio: Calcolo di un valore assoluto di un numero (Schema a blocchi)

- Scriviamo un programma che sia in grado di stampare a monitor il numero letto senza il segno



Esempio: Calcolo di un valore assoluto di un numero (Codifica Pascal)

```
0  program modulo;
1  (* determina il valore assoluto di un numero *)
2  var
3    numero:integer;                    // dato in input
4  begin
5  (* leggi il numero *)
6    write('introduci il numero con segno: ');
7    readln(numero);
8    if numero<0                        // esegui il confronto
9      then numero:=-1*numero;          // cambiagli il segno
10 (* comunica il risultato *)
11   writeln('il valore assoluto del numero letto e'': ');
12   writeln(' ', numero:6);
13   readln;
14 end.
```

Esempio: Lettura di una frazione

Leggiamo una frazione a/b e controlliamo la condizione di esistenza del denominatore b (cioè b diverso da 0). Scriviamo direttamente il codice in linguaggio Pascal:

```
0  program frazione;
1  (* verifica il denominatore di una frazione *)
2  var
3    a,b:integer;                          // dati in input
4  begin
5    write('introduci il numeratore a della frazione a/b: ');
6    readln(a);
7    write('introduci il denominatore b della frazione a/b: ');
8    readln(b);
9    if b=0
10     then
11       begin
12         writeln('attenzione');
13         writeln('il denominatore e'' uguale a 0!');
14       end;
15   readln;
16 end.
```

Inizio/fine if

- Come si fa a dichiarare l'inizio e la fine dell'*if* ?
  - Se il ramo *then* contiene una sola istruzione è *sufficiente* inserire al termine dell'istruzione stessa " ; "
  - Se il ramo *then* contiene più istruzioni è *necessario* inserire "begin…end;" rispettivamente prima e dopo il blocco di istruzioni presenti nel ramo

L'indentamento

È importante guardare come sono state scritte le istruzioni: puoi notare che è stata rispettata la regola di ◀ **indentamento** ▶ per rendere facilmente comprensibile il codice: le istruzioni "interne" sono state spostate a destra di due caratteri e incolonnate tra di loro.

◀ Indentamento È un metodo di scrittura dei programmi in cui, entrando in un nuovo blocco, il testo del programma viene scritto spostato a destra del testo che lo precede di qualche carattere bianco, in modo da evidenziare la struttura annidata dei blocchi. Questa modalità di scrittura è del tutto ininfluente sul significato del programma e viene trascurata dal compilatore, mentre è utilissima per la comprensione del programma stesso da parte dell'utente. ▶

Esercizio: ordinamento

- Ordinare tre numeri in senso crescente:
  - Leggere tre numeri interi
  - Orinarli in senso crescente
  - Scrivere a monitor i tre numeri ordinati

Soluzione: ordinamento

```
0  program ordina;
1  var
2    num1, num2, num3, tempo: integer;
3    min, inter, max: integer;
4  begin
5    num1 :=0;
6    num2 :=0;
7    num3 :=0;
8    write('introduci il primo numero : ');     // leggi il primo numero
9    readln (num1);
10   max := num1;
11   write('introduci il secondo numero: ');    // leggi il secondo numero
12   readln (num2);
13   min := num2;
```

Soluzione: ordinamento

```
13   min := num2;
14   if min > max                                    //scambiali
15     then
16       begin
17         tempo := min;
18         min := max;
19         max := tempo;
20       end;
```

Soluzione: ordinamento

```
21  write('introduci il terzo numero : ');     // leggi il terzo numero
22  readln (num3);
23  inter := num3;
24  if inter > max                             //scambiali
25    then
26      begin
27        tempo := inter;
28        inter := max;
29        max := tempo;
30      end;
```

Soluzione: ordinamento

```
31   if inter < min                                    //scambiali
32     then
33       begin
34         tempo := inter;
35         inter := min;
36         min := tempo;
37       end;
38   // scrivi il risultato
39   writeln('i tre numeri ordinati sono : ', min:4, inter:4, max:4);
40 readln;
41 end.
```

Conclusioni

ABBIAMO IMPARATO CHE...

- Una **condizione logica** è un'espressione che può assumere due soli valori: vero oppure falso (true o false).
- L'istruzione di **selezione semplice** permette di far eseguire al calcolatore alcune istruzioni solamente quando il **valore di una condizione logica** ha esito positivo (o **VERO**); nell'altro caso, cioè quando il risultato ha valore **FALSO**, **non viene eseguita** alcuna istruzione.
- Una **condizione logica** è un'espressione del linguaggio rappresentata da due elementi messi a confronto da un **operatore relazionale**.
- I principali **operatori relazionali** del linguaggio **Pascal** sono: >,< , >=, <=, = , <>.
- Per eseguire più di una istruzione in un ramo di una istruzione di selezione è necessario racchiuderle con **begin ... end**.

Esercizi 1-5 pag. 322

1. Data l'età di una persona, determina se è maggiorenne.
2. Scrivi un programma che legge due numeri e visualizza sullo schermo solo il maggiore di essi: nel caso siano uguali, scrive la frase "i due numeri sono uguali".
3. In una serra si considera normale la temperatura di 18°, sotto i 5° si hanno danni irreparabili, tra i 5° e i 18° vi è una situazione di pericolo: scrivi un algoritmo che, letta la temperatura della serra, indichi lo stato della serra.
4. Leggi un numero e scrivilo a video solo se tale numero è pari.

   Traccia per la soluzione
   Utilizza la divisione intera in modo da ottenere un numero pari dal numero letto in ingresso.
   Quindi confronta questo numero con il numero letto: se sono uguali, il numero letto è pari.
5. Leggi due numeri, fanne la divisione e scrivi a video il risultato solo se è intero utilizzando la funzione **`int`** che applicata a un numero reale dà come risultato la sola parte intera, ad esempio
   `int(4,3)=4` oppure
   `int(3,5)=3.`
   (Suggerimento: utilizza variabili di tipo **`real`**.)

Esercizi 6-12 pag. 322

6 Leggi un numero e scrivilo a video solo se tale numero è divisibile per 5.

7 Leggi un numero e nel caso sia positivo:
a) individua se è pari o dispari
b) individua se è divisibile per 7.

8 Leggi il valore del lato di un cubo e comunica la superficie totale e il volume solo se il valore letto è positivo.

9 Leggi quattro numeri e determina il maggiore e minore e il valore medio.

10 Un supermercato applica uno sconto del 20% sull'importo che supera i 100 euro: scrivi un programma che, leggendo il totale della spesa, calcola l'eventuale importo scontato.

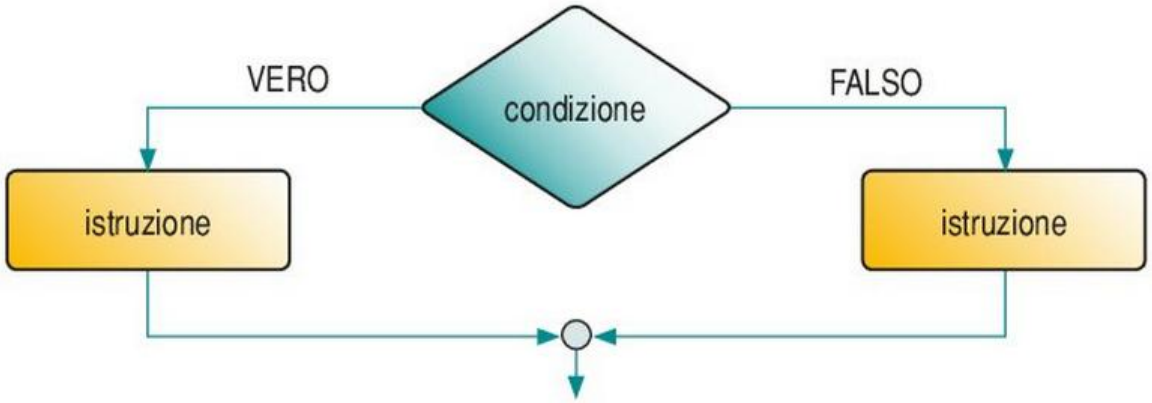
11 Allo stadio il costo del biglietto è gratis fino a 10 anni e sopra i 65, costa 5 euro fino a 18 anni e 10 euro per tutti gli altri: scrivi un programma che legge un numero intero indicante l'età dello spettatore e visualizza l'importo che deve pagare.

12 Scrivi un programma che legge in input due ore espresse in ORE:MIN:SEC e calcola il tempo intercorso tra le due letture. Le due ore possono anche essere di giorni differenti.

La selezione doppia: *if..then..else*

La selezione doppia: *if..then..else*

- Si ha la possibilità di far eseguire al calcolatore alternativamente alcune istruzioni in base al valore dell'istruzione di test (o condizione)



- se la condizione è VERA si eseguono le istruzioni presenti nel ramo sinistro;
- se la condizione è FALSA si eseguono le istruzioni presenti nel ramo destro.

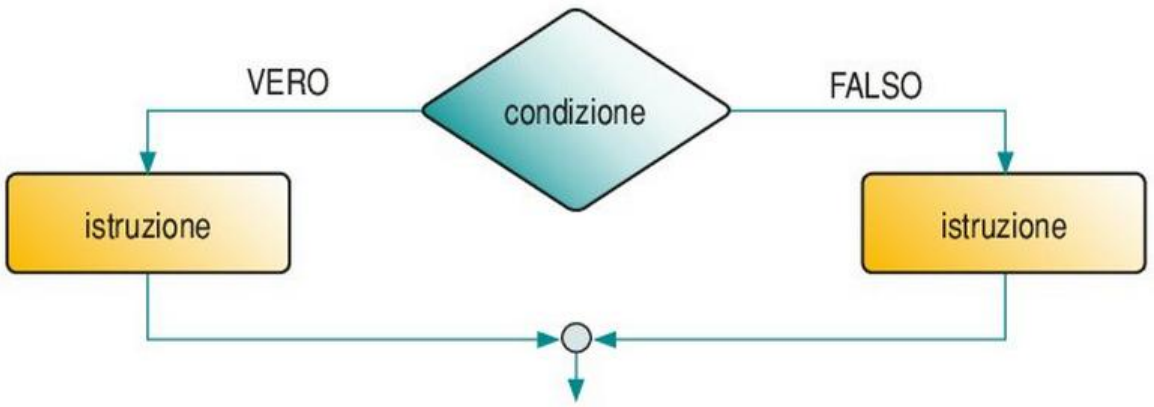
If..then..else in Pascal

```
if(condizione)
  then istruzione
  else istruzione;
```

If..then..else: Codifica Pascal vs schema a blocchi

```
if(condizione)
  then istruzione
  else istruzione;
```

vs



Esempio

- Leggiamo un numero da tastiera e indichiamo se il numero è positivo o negativo

```
0  program positiviNegativi;
1  var
2    numero:integer;                          // dato in input
3  begin
4    write('introduci un numero');
5    readln(numero);                          // leggi il numero
6    write('il numero inserito: ',numero);
7    if numero>0                              // confrontalo con 0
8      then
9        writeln('e'' positivo!')             // NB senza il ;
10     else
11       writeln('e'' negativo!');            // NB qui ci vuole il ;
12     readln;
13 end.
```

Dove va il “;” ?

Con la condizione doppia è facile commettere un errore di sintassi, tipico del linguaggio Pascal. Abbiamo detto che in Pascal è necessario terminare ogni istruzione con il ;. Fai attenzione quindi a non inserire un punto e virgola nella istruzione del ramo then, altrimenti il compilatore capisce che l'istruzione termina in quel punto senza proseguire a leggere il ramo else. Il punto e virgola deve essere messo solo nella istruzione else.

```
0  program positiviNegativi;
1  var
2    numero:integer;                          // dato in input
3  begin
4    write('introduci un numero');
5    readln(numero);                          // leggi il numero
6    write('il numero inserito: ',numero);
7    if numero>0                              // confrontalo con 0
8      then
9        writeln('e'' positivo!')             // NB senza il ;
10     else
11       writeln('e'' negativo!');            // NB qui ci vuole il ;
12   readln;
13 end.
```

Dove va il “;” ? (2)

- Quando le istruzioni presenti nel ramo *then* o nel ramo *else* sono più di una è obbligatorio inserire begin…end rispettivamente prima e dopo il blocco di istruzioni
- Ad eccezione dell’ultimo blocco, quello nell’*else* o l’ultimo della catena di if annidati (che vedremo in seguito), che deve essere inserito tra le parole chiave begin…end;

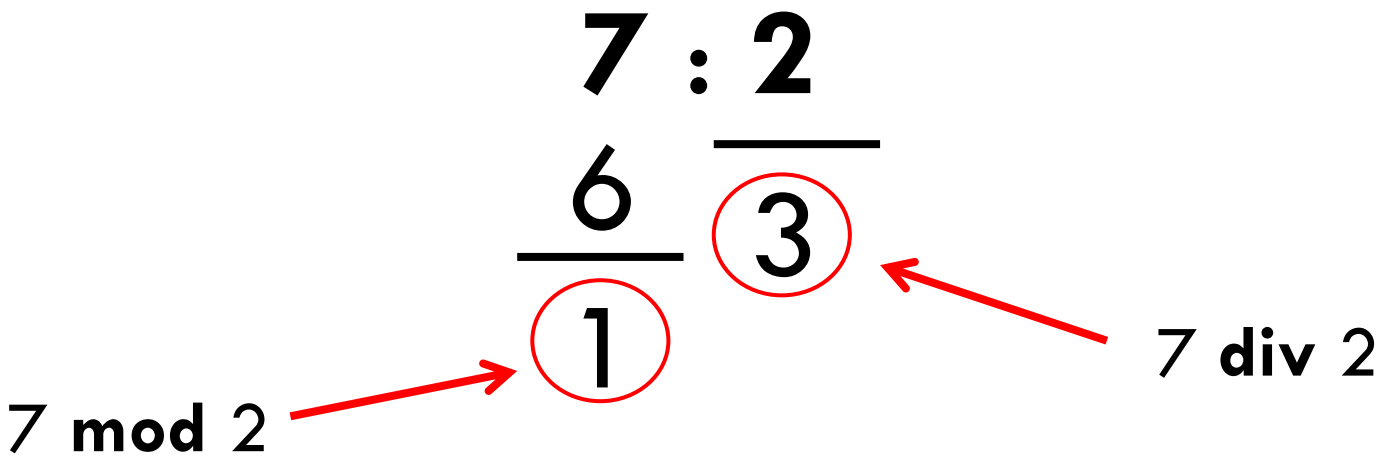
Esempio: Maggiore o minore?

- Leggiamo due numeri e individuiamo il maggiore

```
0  program maggiore;
1  var
2    numero1,numero2:integer;                    // dati in input
3  begin
4    write('introduci due numeri interi');
5    readln(numero1,numero2);                    // leggi i numeri
6    if numero1>numero2                          // confrontali
7      then
8        writeln('il maggiore e'' ',numero1)
9      else
10       writeln('il maggiore e'' ',numero2);
11   readln;
12 end.
```

Operatori *div* e *mod*

Div	Mod
Esegue la divisione tra interi	**Calcola il resto della divisione tra interi**
Il risultato è un itero	Il risultato è un itero
Es: 7 div 2 = 3	Es: 7 mod 2 = 1



Esempio: *div* e *mod*

		div	mod	Commento
7	2	7 div 2 = 3	7 mod 2 = 1	3*2 + 1 = 7
6	2	6 div 2 = 3	6 mod 2 = 0	3*2 + 0 = 6
7	3	7 div 3 = 2	7 mod 3 = 1	2*3 + 1 = 7
6	3	6 div 3 = 2	6 mod 3 = 0	2*3 + 0 = 6
10	3	10 div 3 = 3	10 mod 3 = 1	3*3 + 1 = 10
11	3	11 div 3 = 3	11 mod 3 = 2	3*3 + 2 = 11
20	3	20 div 3 = 6	20 mod 3 = 2	6*3 + 2 = 20
20	6	20 div 6 = 3	20 mod 6 = 2	3*6 + 2 = 20
20	7	20 div 7 = 2	20 mod 7 = 6	2*7 + 6 = 20
30	4	30 div 4 = 7	30 mod 4 = 2	7*4 + 2 = 30
30	7	30 div 7 = 4	30 mod 7 = 2	4*7 + 2 = 30
30	9	30 div 9 = 3	30 mod 9 = 3	3*9 + 3 = 30

Esercizio: pari o dispari?

- Scrivere un programma che sia in grado di determinare se un numero è pari o dispari:
  - Leggere da tastiera un numero intero
  - Verificare se è pari o dispari
  - Comunicare all'utente il risultato
- Usare l'operatore *div*

Soluzione: pari o dispari?

```
0  program pariDispari;
1  var
2    numero:integer;                    // dati in input
3    numeroTroncato:integer;            // variabile di lavoro
4  begin
5    (* leggi il numero *)
6    write('introduci un numero');
7    readln(numero);
8    (* esegui i calcoli *)
9    numeroTroncato:=numero div 2;
10   numeroTroncato:=numeroTroncato*2;
11   (* comunica i risultati *)
12   write('il numero inserito: ',numero);
13   if numero=numeroTroncato
14     then
15       writeln('e'' pari!')
16     else
17       writeln('e'' dispari!');
18   readln;
19 end.
```

Da notare che…

- La condizione scritta nell'istruzione if viene racchiusa tra parentesi tonde:

```
if(numero1=numero2)
```

- Questa notazione sarà **obbligatoria** per le *condizioni composte*.

Esercizio: Pari o dispari (2)

- Esegui l'esercizio precedente utilizzando la funzione ***mod***

Funzione int()

- Applicata a un numero reale restituisce come risultato la sola parte intera:

num	int(num)
3,56	3
56,3445443	56
0,23	0
9,99	9

Esercizio: Pari o dispari (3)

- Esegui l'esercizio precedente utilizzando la funzione ***int()***

Conclusioni

ABBIAMO IMPARATO CHE...

- L'istruzione di **selezione doppia** permette di **far eseguire al calcolatore alternativamente alcune istruzioni** in base al valore della istruzione **condizionale**: se l'**esito è positivo** si eseguono le istruzioni presenti nel **ramo di sinistra**, se invece è negativo si eseguono le istruzioni presenti nel **ramo di destra**.
- Utilizzando l'operatore **div** e la selezione doppia abbiamo individuato se un numero letto è pari o dispari.
- Per eseguire più di una istruzione in un ramo di una istruzione di selezione è necessario racchiuderle con **begin ... end**.

Esercizi1-6 pag. 329

1 Scrivi un programma che legge un numero intero e visualizza sullo schermo il suo triplo se è un numero dispari, il suo doppio se è un numero pari.

2 Leggi un numero e scrivi a video se tale numero è intero oppure è decimale utilizzando la funzione **`int`** che applicata a un numero reale dà come risultato la sola parte intera (ad esempio **`int(4,3)=4`** oppure **`int(3,5)=3`**).

3 Scrivi un programma che legge due numeri e, se sono diversi, ne visualizza il valore medio, se sono uguali il numero stesso.

4 Scrivi un programma che legge un numero e ne calcola la radice quadrata solo se tale numero è positivo utilizzando la funzione **`sqrt(x)`**.

5 Scrivi un programma che legge i coefficienti a e b di una equazione di primo grado $ax + b = 0$ e ne determina la radice.

6 Scrivi un programma che legge i coefficienti *a*, *b* e *c* di una equazione di secondo grado $ax^2 + bx + c = 0$ e ne determina le radici analizzando tutte le possibili situazioni.

Esercizi 7-11 pag. 329

7 Scrivi un programma che legge due numeri e li visualizza in ordine crescente, cioè dapprima il più piccolo e quindi il più grande.

8 Un negozio effettua uno sconto del 10% se il totale speso è inferiore a 500 euro e del 20% se invece è superiore. Scrivi un programma che, inserendo il totale speso, ne calcola lo sconto e visualizza sullo schermo sia lo sconto che l'importo da pagare.

9 Un negozio concorrente effettua uno sconto del 10% sui primi 300 euro di spesa e il 20% sul resto della spesa. Scrivi un programma che, inserendo il totale speso, ne calcola lo sconto e visualizza sullo schermo sia lo sconto che l'importo da pagare.

10 Scrivi un programma che determina fino a che importo conviene fare acquisti nel primo negozio e quando invece è più conveniente il secondo negozio.

11 Un terzo negozio concorrente effettua uno sconto del 50% sul secondo prodotto che compri pagando il primo a prezzo intero: naturalmente si paga a prezzo intero il prodotto più costoso. Scrivi un programma che leggendo i prezzi di due prodotti acquistati calcola il totale da pagare.

Operatori logici: AND, OR e NOT

Variabili booleane

- Una condizione logica è un'istruzione che ha come risultato solo due possibili alternative: VERO o FALSO
- Variabili che possono memorizzare solo i valori VERO (TRUE) o FALSO (FALSE) sono definite di tipo **boolean**

Esempio: utilizzo di una variabile booleana

```
0 program pioggia1;
1 var
2   piove:boolean;                    // dichiarazione
3 begin
4   piove:=TRUE;                      // inizializzazione
5   if piove                          // utilizzo come condizione
6     then
7       write('apri l''ombrello');
8   readln;
9 end.
```

Per migliorare la leggibilità del codice si può anche scrivere:

```
if piove=TRUE                         // scrittura completa
```

Esempio: operatore di disuguaglianza

```
0  program pioggia2;
1  var
2    piove:boolean;                    // dichiarazione
3  begin
4    piove:=TRUE;                      // inizializzazione
5    if piove<>FALSE                   // utilizzo come condizione
6      then
7        write('apri l''ombrello');
8    readln;
9  end.
```

Il risultato è lo stesso del programma precedente. Infatti la condizione viene letta nel seguente modo: *"se piove è diverso da falso"*, *che equivale a dire "se piove è uguale a vero"*.

Ricorda che la scrittura diverso da falso ed è uguale a vero hanno il medesimo significato.

Gli operatori logici

- Le variabili boolean si chiamano così in onore del matematico inglese George Boole
- Boole ha studiato e definito le operazioni possibili su di esse, e sono:
  - La negazione NOT
  - Il prodotto (o congiunzione) AND
  - La somma (o disgiunzione) OR

La negazione NOT

- L'operazione di negazione esegue il cambiamento del valore della variabile.
- Ad esempio eseguendo $X := \texttt{not}(X)$ si ottiene:
  - se X aveva valore VERO, dopo l'istruzione ha valore FALSO;
  - se X aveva valore FALSO, dopo l'istruzione ha valore VERO.

NOT viene utilizzato per invertire il significato della condizione

Esempio: piove o non piove?

```
0  program pioggia3;
1    var
2      piove:boolean;
3    begin
4      piove:=TRUE;
5      if not(piove)
6        then
7          write('non piove')
8        else
9          write('apri l''ombrello: piove vale ',piove);
10     readln;
11 end.
```

L'istruzione poteva anche essere scritta nella forma equivalente completa:
`if not(piove=TRUE).`

La condizione testata è diventata *"se non piove"*, cioè, più precisamente, *"se il valore della variabile* `piove` *non è uguale a TRUE"*.

Prodotto logico AND

- Il prodotto logico tra due variabili ha come risultato il valore VERO solo se entrambe le variabili hanno valore VERO
- L'operatore AND può essere descritto mediante una *tabella di verità*

A	B	A and B
FALSO	FALSO	FALSO
VERO	FALSO	FALSO
FALSO	VERO	FALSO
VERO	VERO	VERO

Esempio: AND

- Analizziamo la seguente espressione e individuiamo quando l'esito dell'espressione ha valore VERO:

$$(A>10)\texttt{and}(A<20)$$

- la prima condizione ha valore VERO quando A è maggiore di 10;
- la seconda condizione ha valore VERO quando A è minore di 20;
- entrambe sono verificate solamente per valori di A compresi tra 10 e 20.

(A>10)and(A<20) = TRUE se e solo se 10<A<20

Esempio: paganti al cinema

- A un cinema i bambini con meno di 10 anni e gli anziani con più di 80 anni non pagano il biglietto.
- Scriviamo un problema che, in base all'età, indichi se si ha diritto all'ingresso gratuito.

Svolgimento: paganti al cinema

```
0  program cinema1;
1  var
2    anni:integer;
3  begin
4    write('quanti anni hai?');
5    readln(anni);
6    if(anni>10)and(anni<80)
7      then
8        write('devi pagare il biglietto')
9      else
10       write('hai l''ingresso gratuito');
11   readln;
12 end.
```

Attenzione: l'espressione (anni>10)and(anni<80) è diversa dall'espressione (anni<10)and(anni>80).
Il valore della variabile anni deve verificare contemporaneamente i due confronti e, nel secondo caso, se un numero è minore di 10 non potrà anche essere maggiore di 80: quindi non sarà mai verificata la condizione complessa.

La somma logica OR

- La somma logica tra due variabili ha come risultato il valore VERO anche se solo uno dei due termini ha valore VERO
- La tabella di verità dell'OR è la seguente:

A	B	A or B
FALSO	FALSO	FALSO
VERO	FALSO	VERO
FALSO	VERO	VERO
VERO	VERO	VERO

L'unico caso in cui il risultato è FALSO è quando entrambe le variabili hanno valore FALSO

Esempio: paganti al cinema (2)

Ripetiamo l'esercizio precedente usando OR

- *A un cinema i bambini con meno di 10 anni e gli anziani con più di 80 anni non pagano il biglietto.*
- *Scriviamo un problema che, in base all'età, indichi se si ha diritto all'ingresso gratuito.*

Soluzione: paganti al cinema (2)

```
0  program cinema3;
1  var
2    anni:integer;
3  begin
4    write('quanti anni hai?');
5    readln(anni);
6    if(anni<10)or(anni>80)
7      then
8        write('hai l''ingresso gratuito')
9      else
10       write('devi pagare il biglietto');
11   readln;
12 end.
```

L'istruzione(anni<10)or(anni>80) dà risultato VERO se una delle due componenti ha valore VERO, cioè se:

- o anni sono inferiori a 10;
- o anni sono superiori a 80.

Chi “conta” di più?

- Relazione d’ordine
  - Il valore VERO è maggiore del valore FALSO:

    VERO>FALSO
- Priorità nell’espressione (senza parentesi)
  - NOT precede l’AND
  - l’AND precede l’OR

1. A or B and C
2. (A or B) and C
3. A or B and not C

In queste espressioni quali operatori vengono eseguiti prima?

Priorità nell'espressione

- Esplicitando le priorità con le parentesi si ha:

1. A or B and C ⟶ 1. A or (B and C)
2. (A or B) and C ⟶ 2. (A or B) and C
3. A or B and not C ⟶ 3. A or [B and (not C)]

Priorità nell'espressione (2)

- Per meglio comprendere la priorità possiamo fare le seguenti sostituzioni:

not	⟶	-
or	⟶	+
and	⟶	*

1. A or B and C	⟶	1. A + B * C
2. (A or B) and C	⟶	2. (A + B) * C
3. A or B and not C	⟶	3. A + B * -C

Conclusioni

ABBIAMO IMPARATO CHE...

- Nei linguaggi di programmazione è possibile definire un tipo di variabile dedicata alla memorizzazione di dati che hanno solo i valori VERO e FALSO (true e false): si tratta delle variabili di tipo **boolean** (o booleane).
- Il linguaggio Pascal mette a disposizione variabili di tipo **boolean**.
- Sulle variabili di tipo **boolean** è possibile effettuare le seguenti operazioni:
  - la negazione **not**;
  - il prodotto o congiunzione **and**;
  - la somma o disgiunzione **or**.
- È definita una priorità di esecuzione per gli operatori logici e relazionali: se non vengono inserite parentesi, **not** precede **and** che precede **or**.

Esercizi pag. 336

1 **L'istruzione `not A and B` ha come risultato VERO se:**

a. A e B sono VERI
b. A = VERO e B = FALSO
c. A = FALSO e B = VERO
d. A e B sono FALSI

2 **L'istruzione `not (A and B)` ha come risultato VERO se:**

a. A e B sono VERI
b. A = VERO e B = FALSO
c. A = FALSO e B = VERO
d. A e B sono FALSI

3 **L'istruzione `not A or B` ha come risultato FALSO se:**

a. A e B sono VERI
b. A = VERO e B = FALSO
c. A = FALSO e B = VERO
d. A e B sono FALSI

4 **L'istruzione `not (A or B)` ha come risultato VERO se:**

a. A e B sono VERI
b. A = VERO e B = FALSO
c. A = FALSO e B = VERO
d. d A e B sono FALSI

Esercizi pag. 336

1 Che risultati dà questo codice?

```
var A,B,C:boolean
  A:=false;
  B:=not(A);
  C:=not(A and B);
  writeln(C);
```

2 Indica il risultato fornito a video dal codice seguente:

```
var A,B,C: boolean;
begin
  A:=false;
  B:=true;
  C:=(not B)or(A and B);
  writeln(C);
```

3 Che risultati dà questo codice?

```
var A,B,C:boolean
  A:=false;
  B:=not(A);
  C:=not(A or not B);
  writeln(C);
```

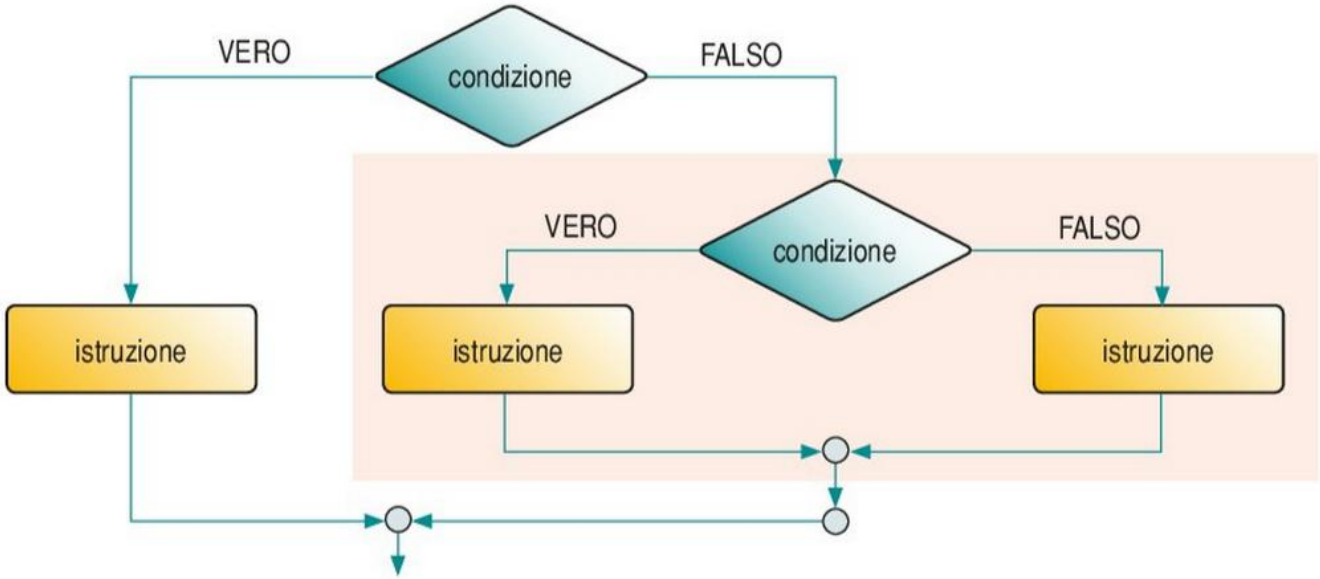
4 Indica il risultato fornito a video dal codice seguente:

```
var A,B,C: boolean;
begin
  A:=false;
  B:=true;
  C:=(not B)and(A or B);
  writeln(C);
```

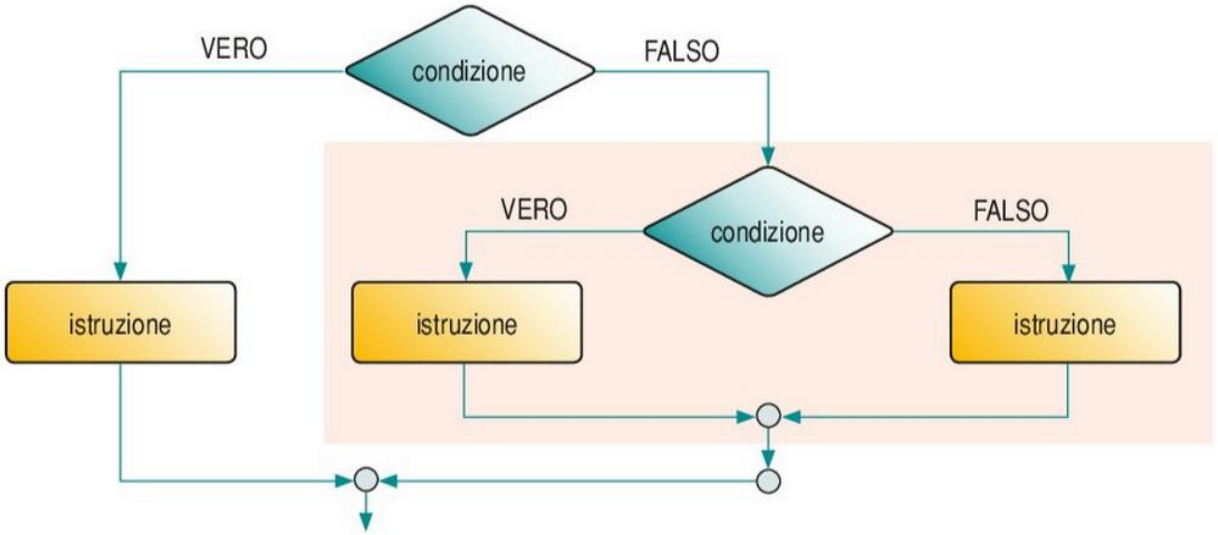
La selezione nidificata (o annidata)

La *selezione nidificata* (o annidata)

- Nei rami delle istruzioni di selezione è possibile eseguire qualunque tipo di istruzione
- Quindi si può anche eseguire un'istruzione di selezione
- Si ottiene dunque una selezione dentro una selezione



Schema a blocchi vs codifica Pascal



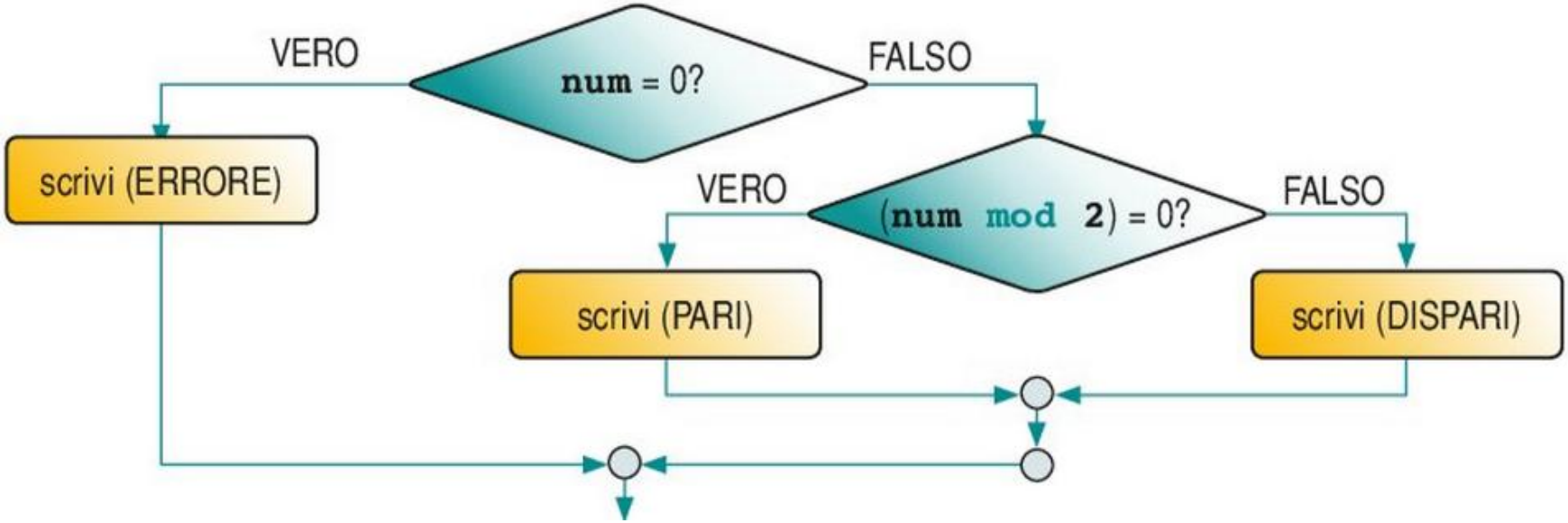

```
if <condizione>
  then
    <istruzione>
  else
    if
      then
        <istruzione>
      else
        <istruzione>;
```

Esempio: Pari o dispari?

- Riscriviamo il programma che individua se un numero è pari o dispari dopo aver verificato che il numero letto sia diverso da 0

- Usiamo la funzione ***mod*** per controllare il resto della divisione per 2 (num mod 2)
  - Se è uguale a 0 allora il numero è *pari*
  - Se è uguale a 1 allora il numero è *dispari*

Soluzione: diagramma a blocchi

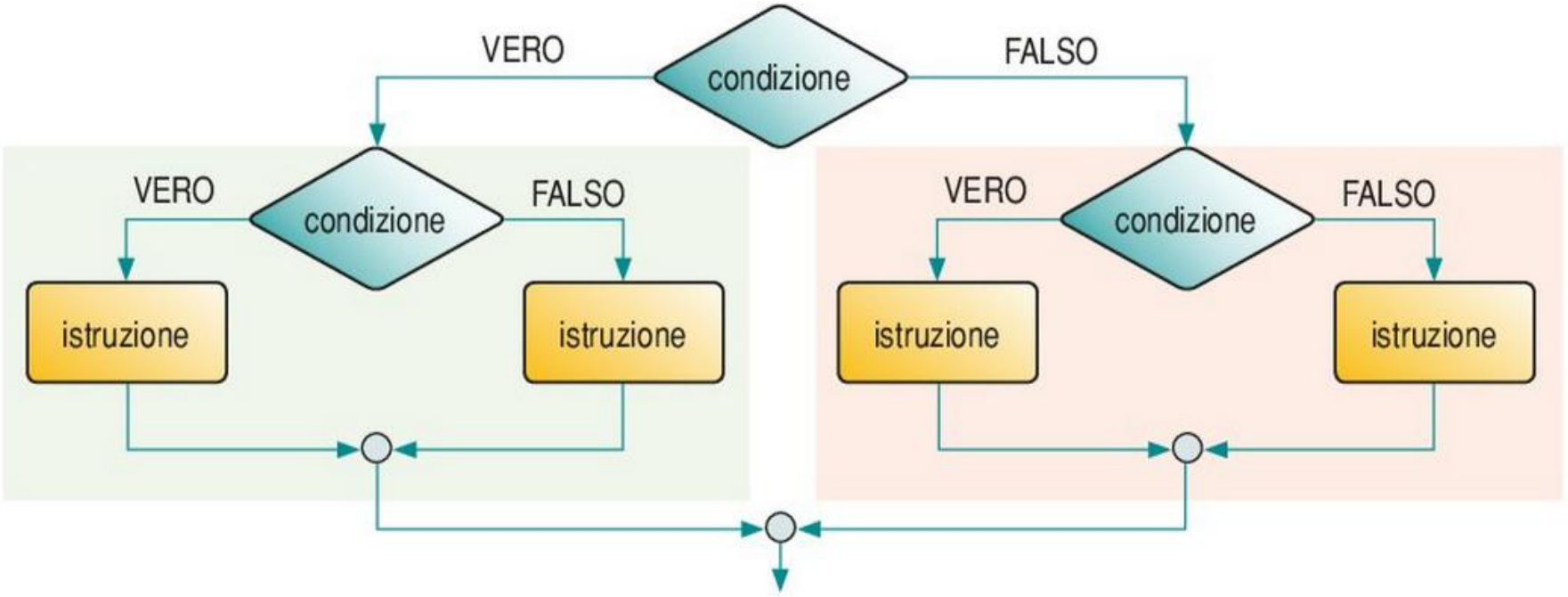


Soluzione: codifica Pascal

```
0  program pariDispari2;
1  var
2    numero:integer;
3  begin
4    write('inserisci un numero: ');
5    readln(numero);
6    if(numero=0)
7      then
8        writeln('numero errato')
9      else
10       if(numero mod 2)=0 // trova il resto
11         then
12           writeln('il numero e'' pari')
13         else
14           writeln('il numero e'' dispari');
15   readln;
16 end.
```

Selezione nidificata in entrambi i rami

- Possiamo inserire un'istruzione di selezione anche in entrambi i rami (sia in *then*, sia in *else*)



Selezione in entrambi i rami: codifica Pascal

- Lo schema precedente tradotto in Pascal diventa:

```
if <condizione>
  then
    if
      then
        <istruzione>
      else
        <istruzione>;
  else
    if
      then
        <istruzione>
      else
        <istruzione>;
```

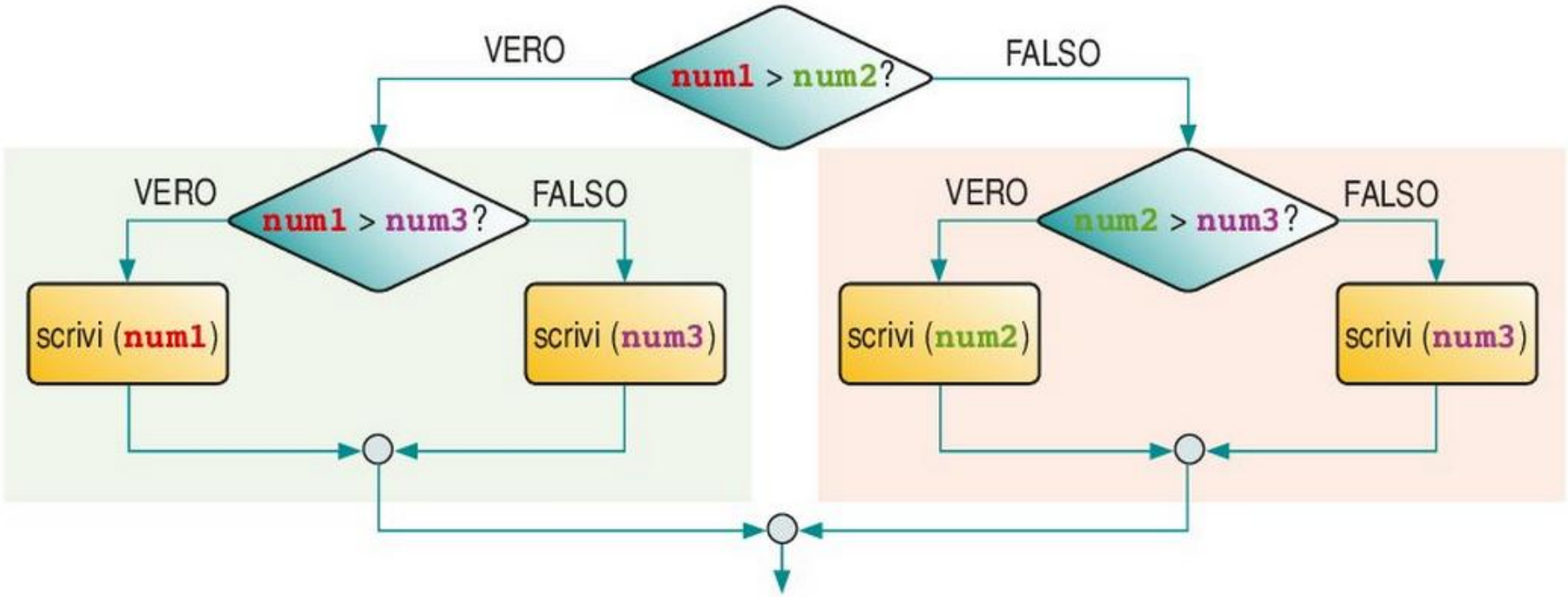
L'istruzione esterna prende il nome di selezione al primo livello, mentre quelle interne sono dette selezioni al secondo livello.

Questa regola vale anche per le condizioni interne, è cioè possibile ripetere lo stesso procedimento fino ad arrivare a condizioni al terzo livello, quarto livello ecc.

Esempio: Maggiore di tre numeri

- Leggiamo tre numeri da tastiera e individuiamo il maggiore

Soluzione: schema a blocchi



Soluzione: Codice Pascal

```
0  program maggiore2;
1  var
2    num1,num2,num3:integer;
3  begin
4    write('inserisci tre numeri separati da spazio: ');
5    readln(num1,num2,num3);
6    if(num1>num2)                    // trova maggiore tra numero1 e numero2
7      then                           // il maggiore è numero1
8        if(num1>num3)                // trova maggiore tra numero1 e numero3
9          then                       // il maggiore di tutti è numero1
10           writeln('il più grande e'' ',num1)
11         else                       // il maggiore di tutti è numero3
12           writeln('il più grande e'' ',num3)
13     else                           // il maggiore è numero2
14       if(num2>num3)                // trova maggiore tra numero2 e numero3
15         then                       // il maggiore di tutti è numero2
16           writeln('il più grande e'' ',num2)
17         else                       // il maggiore di tutti è numero3
18           writeln('il più grande e'' ',num3);
19   readln;
20 end.
```

Esempio: Orafo imbroglione

- Scoprire la moneta più leggera tra le quattro prodotte da un "*orafo imbroglione*" che invece di realizzarle tutte dello stesso peso sottrae una parte di oro da una di esse.
- Noi per individuarla abbiamo a disposizione solo tre pesate.

Soluzione: orafo imbroglione

```
0  program orafo;
1  var
2    moneta1, moneta2, moneta3, moneta4: integer;
3  begin
4    write ('Inserisci peso della moneta 1 : ');
5    readln (moneta1);
6    write ('Inserisci peso della moneta 2 : ');
7    readln (moneta2);
8    write ('Inserisci peso della moneta 3 : ');
9    readln (moneta3);
10   write ('Inserisci peso della moneta 4 : ');
11   readln (moneta4);
12   if (moneta1+moneta2<moneta3+moneta4)
13     then
14       if (moneta1<moneta2)
15         then
16           writeln(' la moneta piu'' leggera e'' la prima ')
17         else
18           writeln(' la moneta piu'' leggera e'' la seconda ')
19     else
20       if(moneta3<moneta4)
21         then
22           writeln(' la moneta più leggera e'' la terza ')
23         else
24           writeln(' la moneta più leggera e'' la quarta ');
25   readln;
26 end.
```

Esercizi 1-7, pag. 344

1 Scrivi un programma che legge due numeri ed effettua lo scambio dei valori di due numeri letti **num1** e **num2** se il valore del primo è maggiore del valore del secondo.

2 Scrivi un programma che legge tre numeri indicando se sono stati introdotti numeri uguali.

3 Scrivi un programma che legge tre numeri da tastiera e indica se costituiscono una terna pitagorica, cioè se $x^2 + y^2 = z^2$.

4 Scrivi un programma che legge quattro numeri da tastiera e trova il minore e il maggiore.

5 Scrivi un programma che legge un numero *N* e, se è diverso da 0, calcola l'area A e il perimetro P del quadrato avente tale numero come lato, segnalando a video il caso in cui l'area ha valore inferiore del perimetro, cioè quando A < P.

6 Scrivi un programma che, leggendo il valore dei lati di un triangolo, ne determini la classificazione.

Traccia per la soluzione
Prima si determina se il triangolo è equilatero: se non siamo in questa situazione si verifica se è isoscele oppure scaleno.

7 Leggi un numero che corrisponde al peso in kg di patate acquistate al mercato: sapendo che ogni sacchetto può contenere al massimo 5 kg di patate e una cassetta fino a 30 kg, determina il numero di sacchetti (massimo 4) o di cassette (massimo 2) necessari per portare a casa la spesa.

Esercizi 8-11, pag. 344

8 Un supermercato effettua lo sconto in base all'importo totale della spesa.
La tabella sconti è:

Importo spesa	% sconto
Spesa < 50 €	5%
Spesa < 65 €	6%
Spesa < 80 €	7%
Oltre 100 €	10%

Inserendo il prezzo totale della spesa visualizza sullo schermo l'importo dello sconto e il costo netto.

9 Il costo del biglietto di un traghetto viene calcolato operando una distinzione tra autovetture e camion, ripartendo ulteriormente i veicoli per cilindrata.

Autovetture		Camion	
Fino a 1000 cc	20 €	Fino a 2000 cc	40 €
Fino a 2000 cc	30 €	Fino a 3000 cc	50 €
Oltre	40 €	Oltre	100 €

Scrivi un programma che permetta di conoscere il costo del biglietto in ogni situazione.

10 Modifica il programma dell'orafo imbroglione facendo in modo di controllare se solo una moneta ha peso inferiore alle altre, cos' da validare il risultato calcolato.

11 Modifica il problema dell'orafo imbroglione considerando di individuare la moneta più leggera tra 10 di peso uguale: realizza un programma che effettua anche i controlli sui dati inseriti in modo da essere certi di avere una sola soluzione.