



**UNIVERSITÀ
DEL SALENTO**

Infomatica

**Corso di Laurea in Scienze Biologiche
Dip.to di Scienze Tecnologie Biologiche ed Ambientali
(cfu 6)**

Prof. Salvatore Mancarella

salvatore.mancarella@unisalento.it

Linguaggio di programmazione



Python

Le applicazioni (Programmi)

Anche i programmi sono interpretati attraverso un'architettura a strati, composta da tre moduli funzionali (sottosistemi) concettualmente indipendenti tra loro:

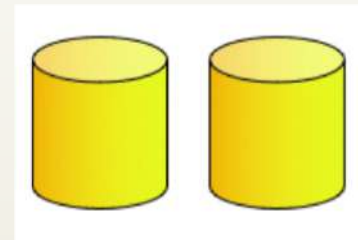
- **Interfaccia Utente (IU)** – Acquisisce i dati e i comandi immessi in *input* dall'utente, e restituisce in *output* i risultati dell'elaborazione.



Le applicazioni (Programmi)

Anche i programmi sono interpretati attraverso un'architettura a strati, composta da tre moduli funzionali (sottosistemi) concettualmente indipendenti tra loro:

- **Logica Applicativa (LA)**– Implementa gli algoritmi specifici per l'elaborazione dei dati e delle informazioni alla base dell'applicazione.
- **Gestione Dati (GD)**– Si occupa della memorizzazione dei dati e ottimizza i metodi per recuperarli, in modo da rendere il più efficiente possibile il loro reperimento e utilizzo.



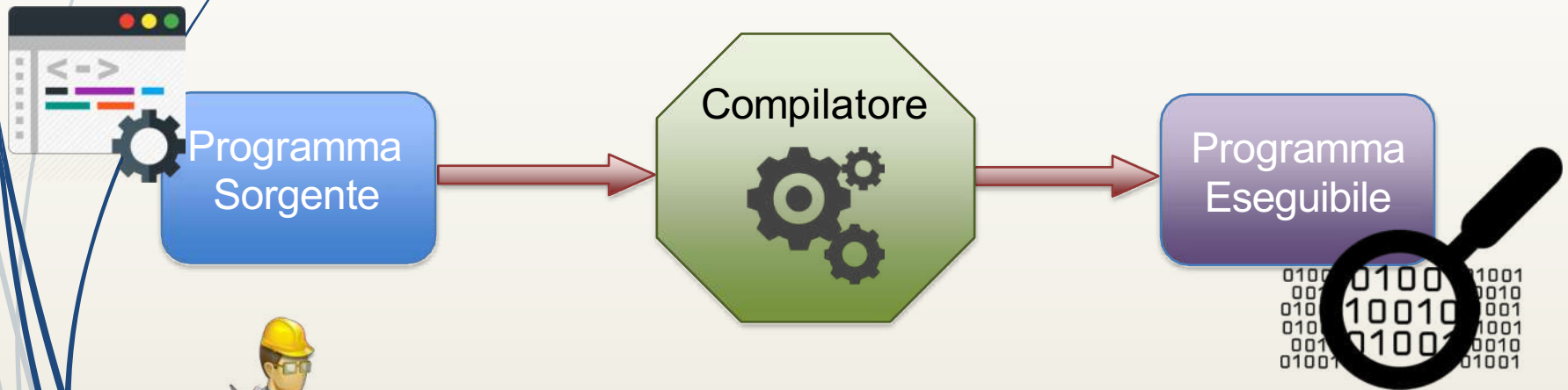
Le applicazioni (Programmi)

Un **programma** è una sequenza finita di istruzioni che, eseguite da un calcolatore elettronico (secondo la logica definita dal programma stesso), produce un'elaborazione su dei dati in ingresso per arrivare a produrre dei dati in uscita, che sono appunto il risultato di questa elaborazione.



Codice sorgente

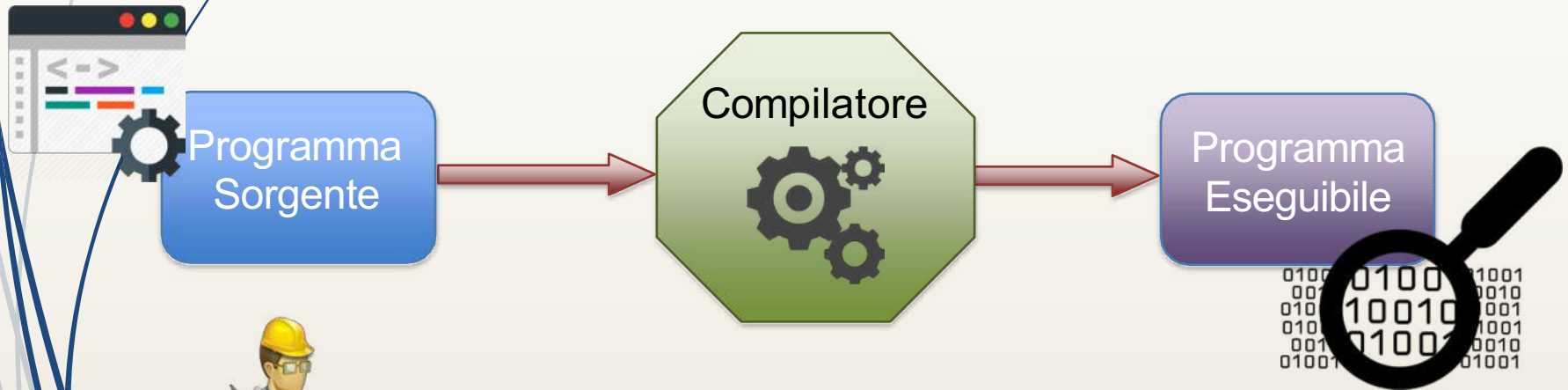
- Quando si realizza un software, il programmatore detta le istruzioni al computer tramite la scrittura di un codice sorgente in un determinato **linguaggio di programmazione**.



Visione lato sviluppatore

Codice sorgente

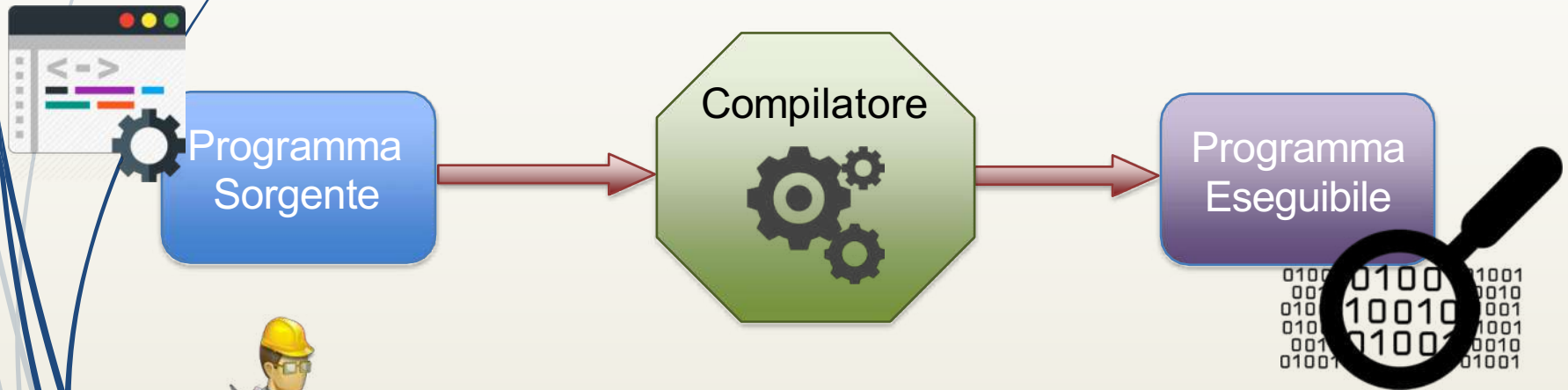
- ▶ Il codice sorgente però non è direttamente eseguibile dalla CPU (Central Processing Unit), è necessario “tradurlo” in linguaggio macchina, ossia un file binario contenente microistruzioni gestibili dal processore.
- ▶ Diverse architetture di processori prevedono linguaggi macchina differenti



Visione lato sviluppatore

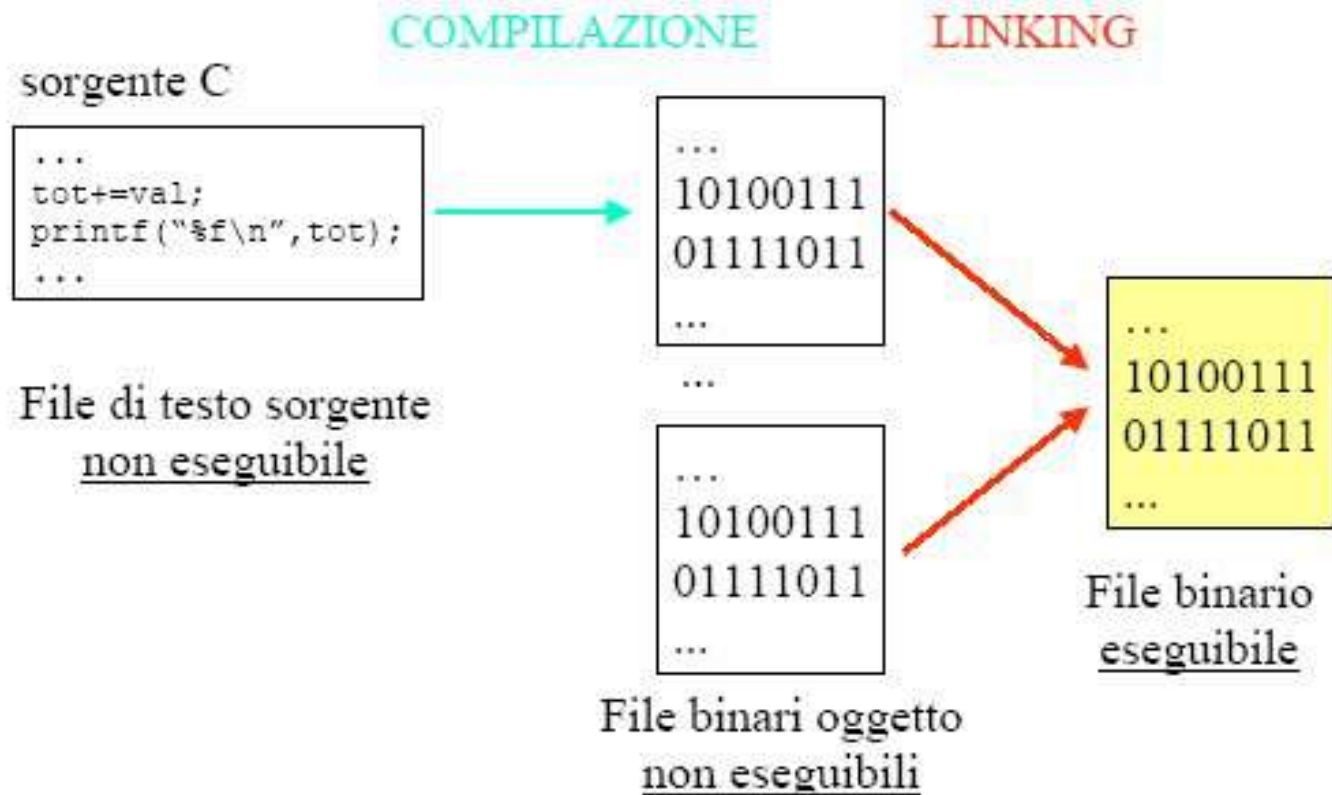
Tipi di linguaggio di programmazione

- **Interpretato** il programma sorgente può essere eseguito direttamente dal calcolatore (interpretati),
- **Compilato** deve essere tradotto da un apposito compilatore (compilati) in linguaggio macchina per poter essere eseguito, ovvero per poter essere trasformato in un programma eseguibile.



Visione lato sviluppatore

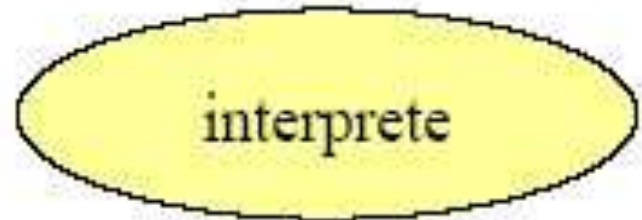
Linguaggio compilato



Linguaggio interpretato

Codice sorgente

```
#!/bin/tcsh -f  
set st=1  
while ( $st != 0 )  
dwl  
...
```



utente

Componenti di un algoritmo

- ➔ **DATI:** sono gli oggetti su cui dobbiamo operare
- ➔ **ISTRUZIONI:** sono le attività che devono essere svolte



I dati

- Ad ogni dato è associato un nome che lo identifica in modo univoco (es. lato, codice, imposta, ecc).
- I dati possono essere classificati secondo il modo in cui interagiscono con l'elaboratore:
 - **Input** (forniti dall'esterno, es. prezzo, quantità, aliquota)
 - **Output** (comunicati all'esterno, es. totale fattura)
 - **Lavoro** (dati di lavoro, es. IVA).



I dati

A seconda degli oggetti che rappresentano, possono essere:

- **Numerici**

- Interi (es: il numero di abitanti – 1,2,3..)
- Reali (es: la temperatura 27,5° 28,6°)

- **Alfanumerici o stringhe**

- Codice fiscale, indirizzo, ecc..





Tipologia di dati

TIPO	ESEMPIO	NOME ESTESO
float	41.77	floating point
int	88	integer
str	"Cate"	string
bool	True	boolean

Dichiarazione delle variabili

- ▶ Quando creiamo una nuova variabile la stiamo **dichiarando** e per dichiarare variabili in Python basta fissarne il nome, seguito da un valore. Ecco alcuni esempi:

```
name = "Cate"  
age = 31  
city = "Rotterdam"  
married = False  
count = 105.00
```

Dichiarazione delle variabili

- Ricollegandoci sempre a "**Python è dinamico**", anche se una variabile nasce stringa non è detto che debba morire tale:

```
name = "Cate"  
name = 81  
# Prima name è una stringa  
# e poi diventa un numero
```

- Anche se Python lo permette non è detto che **tu debba scambiare i tipi continuamente**. Una variabile che si chiama "name" non ha alcun motivo per diventare un numero.

Dichiarazione delle variabili

- Vale la pena chiarire anche che **Python considera come variabili tutti i nomi che nel programma non sono contenuti tra virgolette**. Questo significa che il seguente esempio:

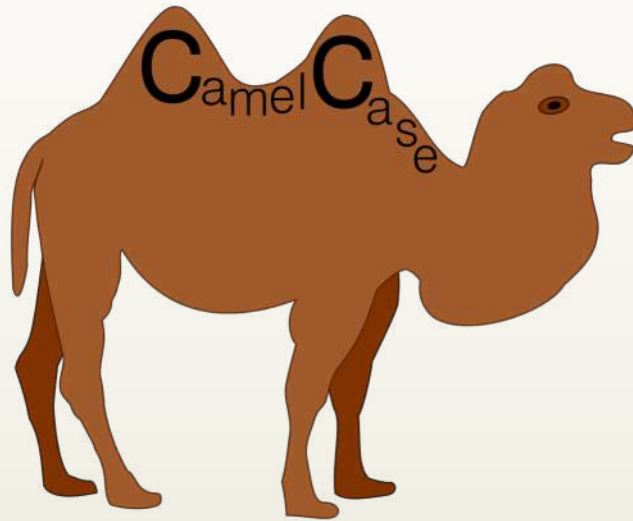
```
name = "Liz"
```

- si legge come **stringa Liz assegnata alla variabile name**. Questo codice invece significa tutt'altro e non è neanche valido:

```
"name" = "Liz"
```

Come scrivere una variabile

- Facciamo qualche esempio. In alcuni linguaggi di programmazione le variabili vengono scritte con la forma cosiddetta camel case:



```
# Non farlo in Python  
myVariable = "Liz"
```

Come scrivere una variabile

```
# Non farlo in Python  
myVariable = "Liz"
```

- in Python questa forma è da evitare perché non è né comune né accettata dalla community.
- La forma corretta invece è:

```
# ok!  
my_variable = "Liz"
```

Separa quindi con un underscore i nomi lunghi

Come scrivere una variabile

- ▶ Un'altra cosa da evitare è il maiuscolo (a parte per un caso particolare):

```
# Non farlo in Python  
MYVARIABLE = "Liz"
```

- ▶ Un altro errore comune ma comprensibile che molti principianti commettono è quello di non avere **consistenza** nello scrivere i nomi delle variabili.
- ▶ Le variabili devono essere parlanti

```
# NON CI SIAMO  
Nome = "Cate"  
ETà = 31  
CITTA = "Rotterdam"  
Married = False  
count = 105.00
```



Come scrivere una variabile

- Questi piccoli accorgimenti migliorano la **leggibilità del codice** e rendono ciò che scrivi **familiare anche agli altri sviluppatori Python**.





I commenti



- A questo punto ti starai chiedendo cosa sono tutti questi cancelletti # nel codice.
- Il **simbolo #** in Python è un **commento**, e serve per lasciare indicazioni nel codice, sia per gli altri sviluppatori, sia per noi stessi.
- I commenti **vengono ignorati dal motore Python** e quindi sono molto utili per **annotare e spiegare il codice**, eliminare temporaneamente istruzioni, e così via.
-

I commenti

- ▶ Ci sono due tipi di commento in Python,
 - ▶ il commento a **linea singola**:

```
# ciao, sono un commento  
name = "Amy"
```

- ▶ ed il commento **multilinea**:

```
"""  
This is a  
multi line  
comment  
"""  
name = "Andrea"
```

Le istruzioni

Le istruzioni Analizziamo i tre tipi di istruzioni fondamentali:

- **Letture:** attraverso la quale si assegna ad una variabile un valore tramite digitazione sulla tastiera del PC;
- **Scrittura:** permette di visualizzare tramite video o supporto cartaceo, un messaggio o il valore di una variabile;
- **Assegnazione:** permette di attribuire un valore ad una variabile.
 - Es: $A=3$; oppure $A=B+3$;



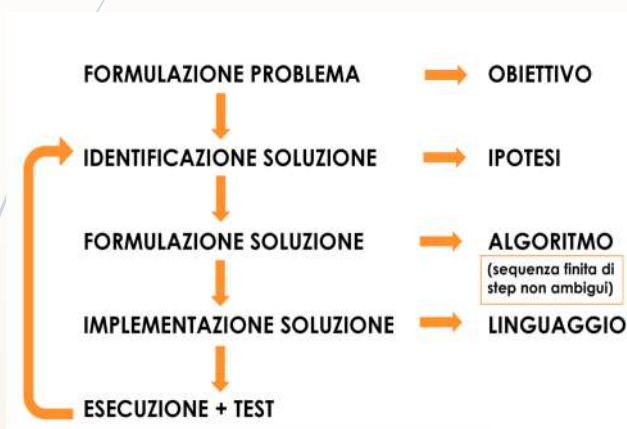


Step pensiero computazionale



Scomposizione

Problema 1



Problema 2



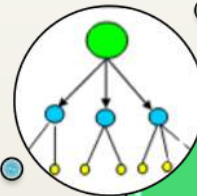
pensiero computazionale



Verifica costante



Trovare la soluzione



Scomporre il problema in sottoproblemi



Analizzare il problema e determinare gli obiettivi

Algoritmi



Algoritmi

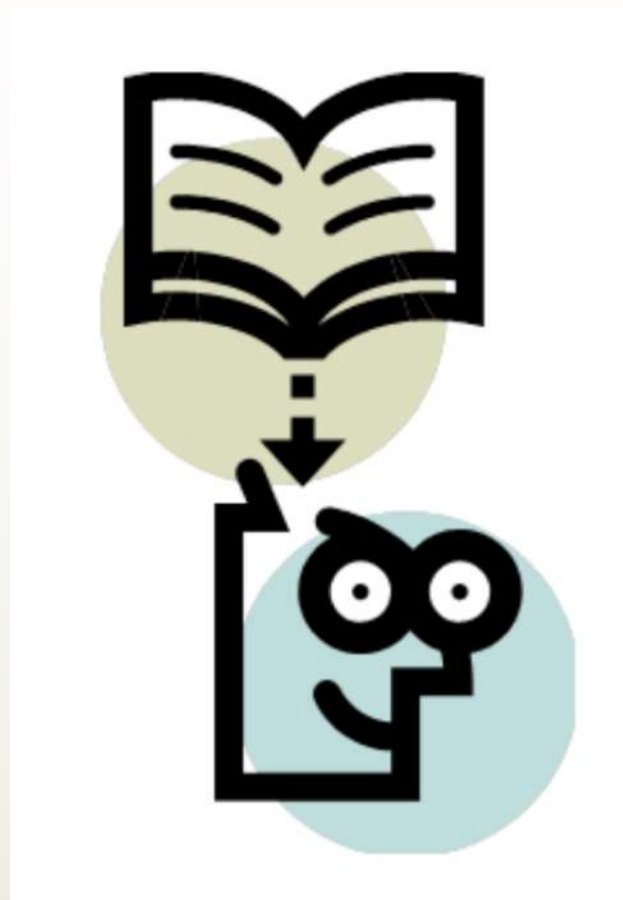
Ogni giorno siamo chiamati a risolvere dei problemi, per es:

1. Decidere se riconciliarci con l'amico con cui abbiamo litigato;
2. Calcolare l'area di un rettangolo, note la base e l'altezza;
3. Stabilire quale squadra di calcio vincerà il campionato, ecc.

Noi cercheremo in questa sede di esaminare soltanto i problemi del secondo tipo, poiché solo tali problemi sono risolvibili mediante algoritmi



Cos'è un Algoritmi



Cos'è un Algoritmi

- ➔ E' una sequenza finita di passi (istruzioni), che devono essere eseguiti secondo un ordine prefissato per individuare la soluzione



Algoritmi

E' però necessario analizzare prima il problema e poi trovare una strategia per risolverlo.

- ▶ Mentre la macchina può essere utilizzata solo per spostarsi,
- ▶ il COMPUTER è in grado di risolvere problemi di natura diversa.



Cos'è il COMPUTER?

E' un sistema di elaborazione che trasforma le informazioni che riceve in ingresso nei risultati che vogliamo ottenere

**Dati di
input**



computer



**Dati di
output**

Caratteristiche dell' algoritmo

L'algoritmo deve essere:

- **Finito:** composto da un numero finito di passi
- **Deterministico:** a fronte degli stessi dati in input deve produrre gli stessi risultati
- **Non ambiguo:** i passi che lo compongono devono essere interpretati in modo univoco dall'esecutore
- **Generale:** deve fornire la soluzione per tutti i problemi appartenenti ad una certa classe



I dati

In base alla possibilità di cambiare il valore durante l'esecuzione dell'algoritmo, distinguiamo:

- **Costanti**, il cui valore rimane immutato nel tempo (es. aliquota IVA);
- **Variabili**, il cui valore può cambiare nel tempo (es. prezzo di un prodotto)



Rappresentazione degli algoritmi

- **Diagramma a blocchi o di flusso o Flow-chart:** è il metodo più usato e si basa sull'uso di simboli a cui corrispondono delle precise operazioni.
- **Pseudocodifica :** utilizza un linguaggio speciale per descrivere le istruzioni da eseguire





➤ <https://colab.research.google.com/>

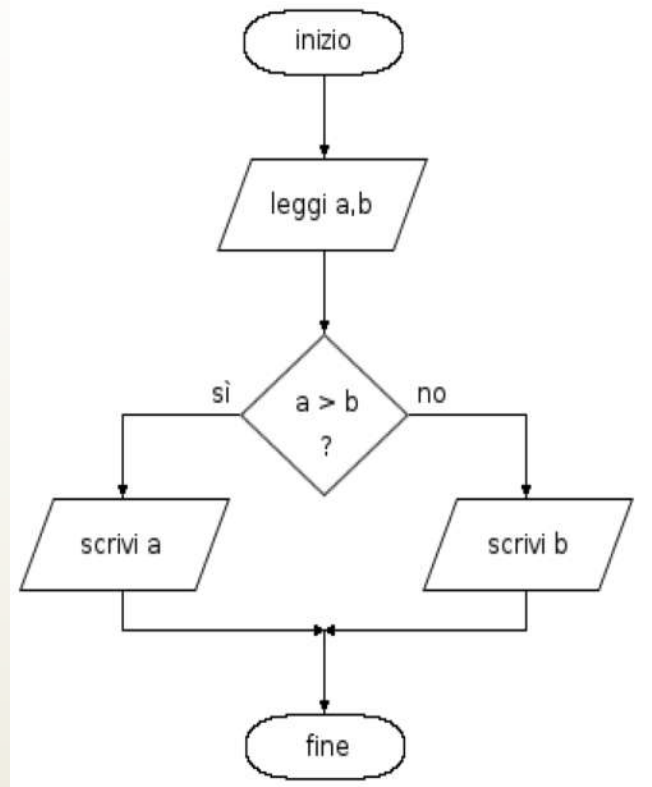


La sequenza

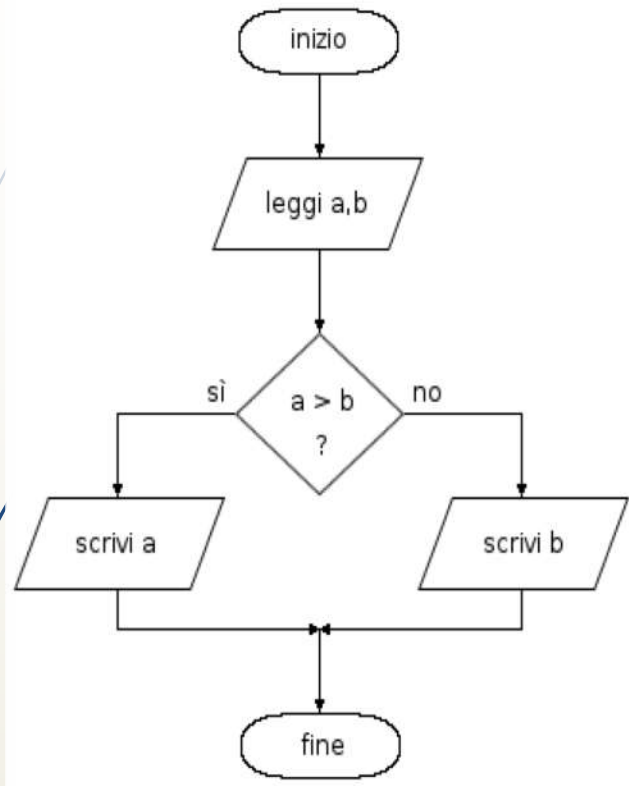


Esercitazione n°1

- Dati due valori determinare il valore più grande



Codice Python



```
a = int(input('Inserisci un numero: A'))
b = int(input('Inserisci un numero: B'))
if b < a: # se il numero è negativo
    print('Il valore maggiore è A = ', a)
else:
    print('Il valore maggiore è B = ', b)
```



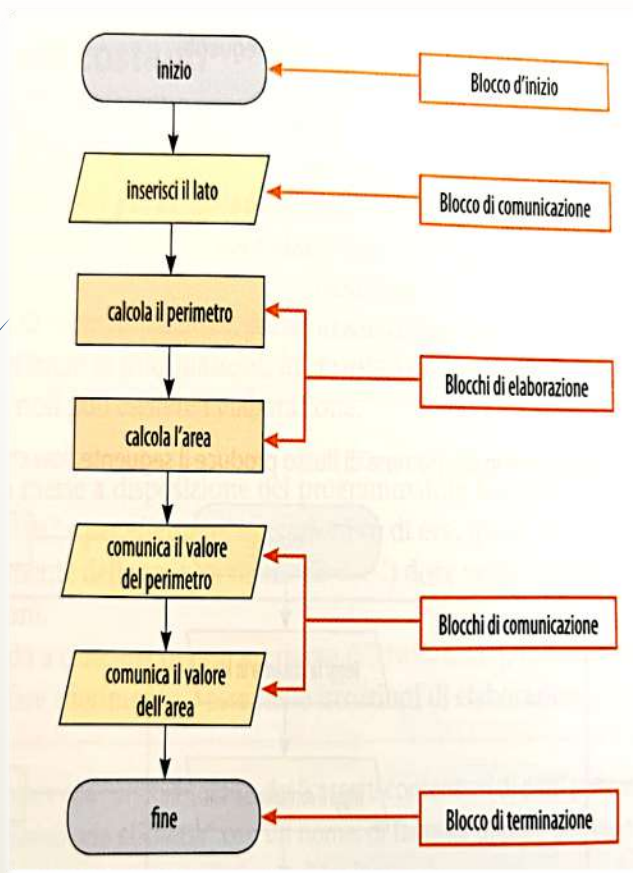


Esercitazione 2



1. Descrivere l'algoritmo che elenca le operazioni necessarie per calcolare il perimetro e l'area di una di un quadrato letto la misura del lato.
2. Descrivere l'algoritmo che calcoli la percentuale di sconto del valore di un determinato bene.
3. Date la temperatura di tre stanze differenti calcolare la temperatura media.
4. Dato il numero di abitanti di un Comune di due anni fa e la popolazione attuale. Calcolare se il comune ha avuto un aumento o una diminuzione della popolazione e la rispettiva percentuale.
5. Descrivere l'algoritmo che calcoli il valore scontato se il prezzo è maggiore di 50 si applica il 5% di sconto, altrimenti il 2%.

Soluzione Es. n° 2.1



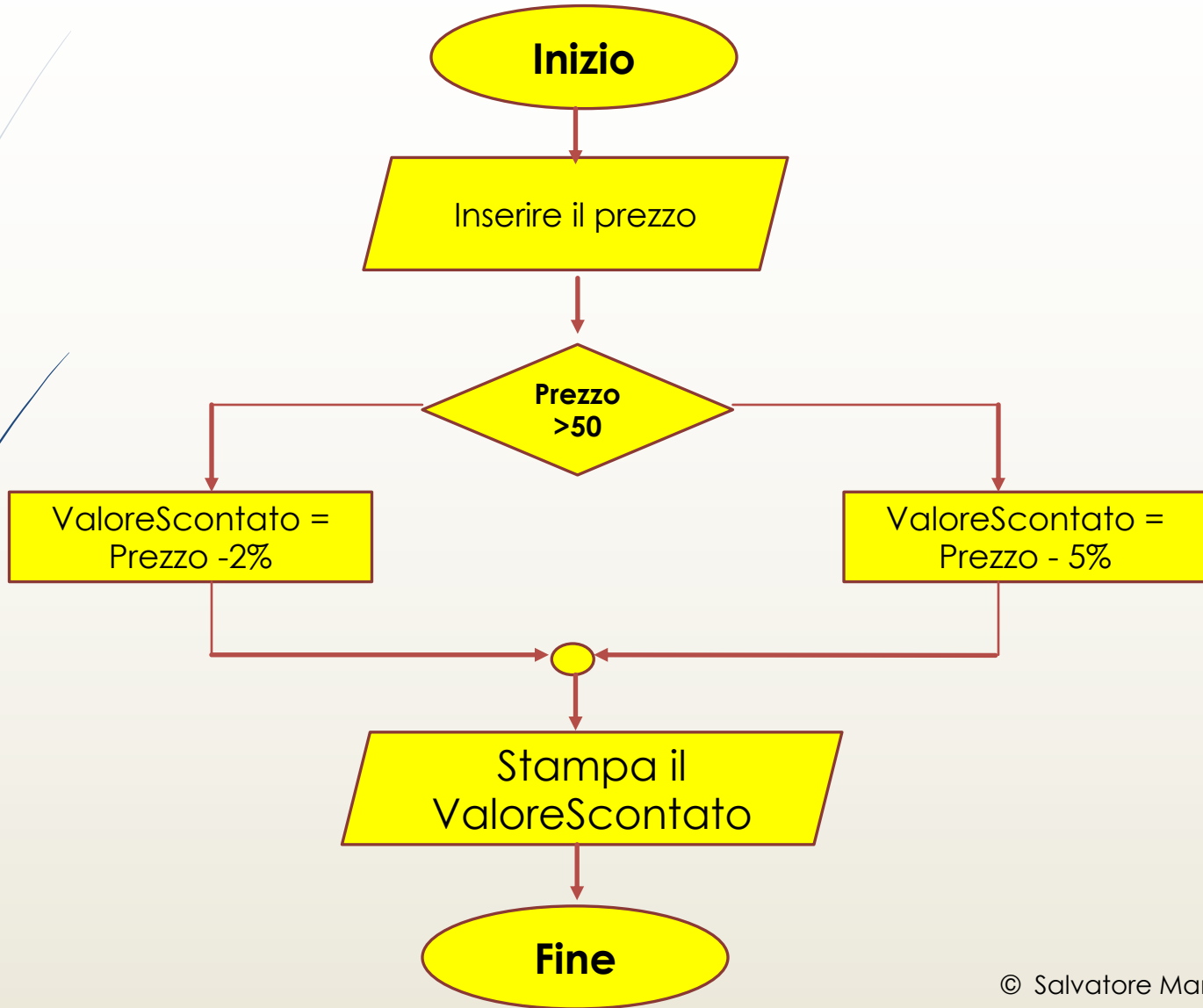
```
lato = int(input('Inserisci il lato del quadrato'))
perimetro = lato*4
area = lato*lato
print('Il valore del perimetro è =', perimetro)
print('Il valore dell'area è =', area)
```



Analisi del problema

- Descrivere l'algoritmo che calcoli il valore scontato se il prezzo è maggiore di 50 si applica il 5% di sconto, altrimenti il 2%.
 - Input → Prezzo
 - Output → Valore Scontato
 - Calcolo → Lo sconto da applicare

Soluzione Es. n°2.5



Soluzione Es. 2.5

➤ Codice Python

```
prezzo = float(input('Inserisci il prezzo'))
valore_scontato = 0.0
if prezzo < 50: # se il prezzo è si applica uno sconto del 5%
    valore_scontato=prezzo - prezzo*5/100
else:
    valore_scontato=prezzo - prezzo*2/100
print('Il valore scontato è = ', valore_scontato)
```



Esercitazione 3

Devi organizzare un viaggio con i tuoi amici, devi esaminare i costi, confrontando le spese del viaggio in autobus o in treno per raggiungere la meta. Sulla base dei calcoli effettuati, devi decidere qual è il mezzo di trasporto più conveniente

Analisi del problema

➤ Dati di input

- Numero dei partecipanti → **NP**
- Costo del treno per persona (andata e ritorno) → **CTP**
- Costo complessivo dell'autobus → **CA**

➤ Dati di output

- Scelta più conveniente tra treno e autobus

**Dati di
input**



computer



**Dati di
output**

Soluzione del problema

► Dati di input

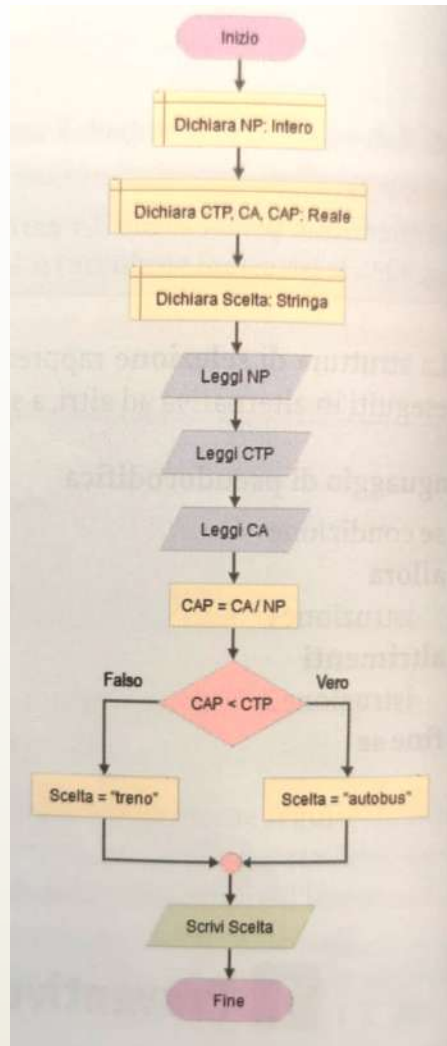
1. Acquisire il numero dei partecipanti → **NP**
2. Acquisire il costo del treno per persona → **CTP**
3. Il costo complessivo dell'autobus → **CA**

► Dati di calcolo

1. **CAP** → Costo dell'autobus per persona = **CA/NP**

► Dati di output

- **SE** il CAP è minore di CTP
 - la scelta è **Autobus**
- **ALTRIMENTI**
 - la scelta è **Treno**





Esercitazioni n°3



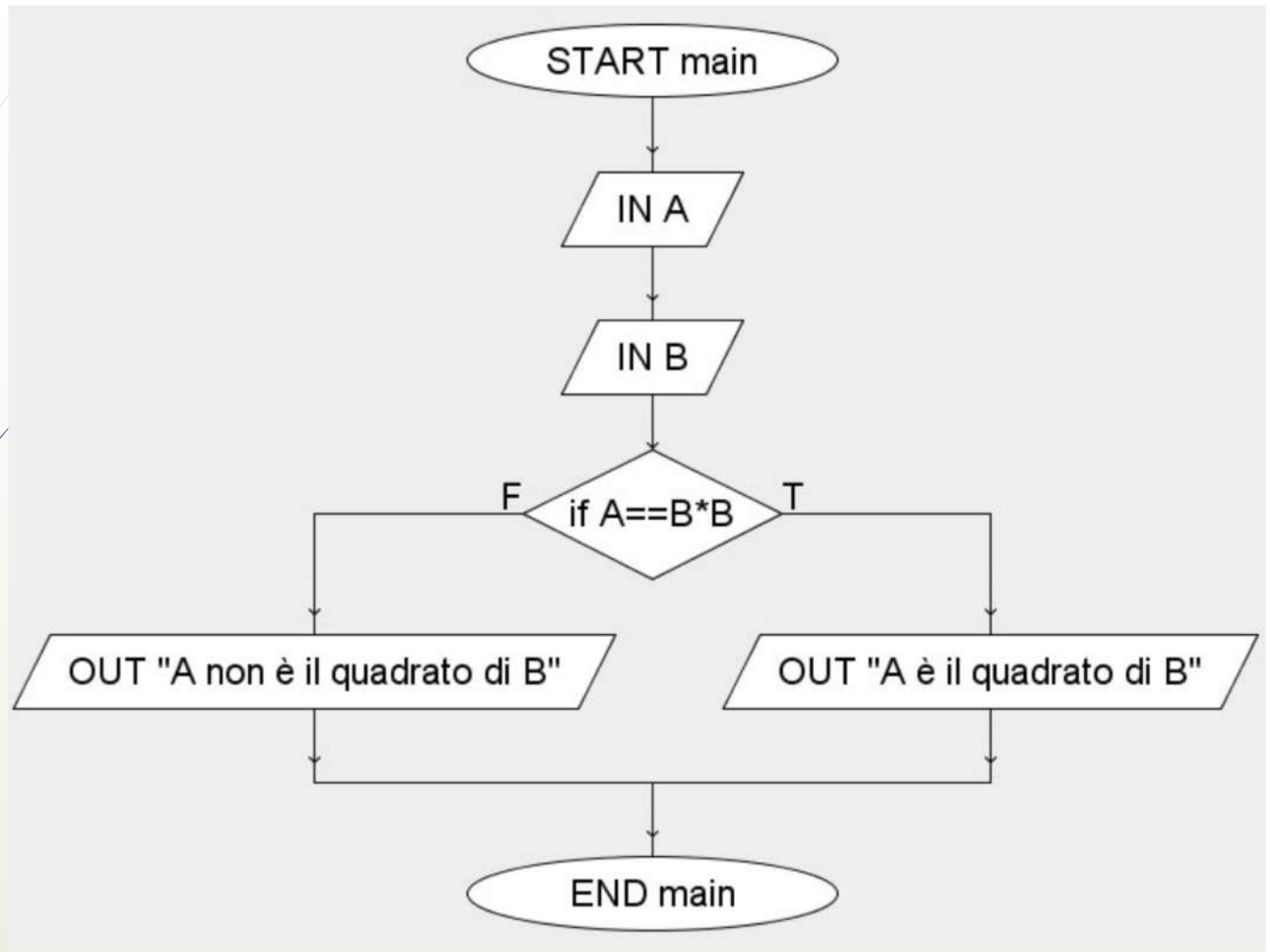
1. Dati due numeri A e B verificare se A è il quadrato di B
2. Data una temperatura **T**, visualizzare se si tratta di una temperatura: “sotto lo zero”, “uguale a zero” o “sopra lo zero”. Visualizzare il messaggio in output.
3. Date le dimensioni di due rettangoli calcolarne l'area e determinare quale dei due ha la superficie maggiore.
4. Realizziamo un algoritmo per il calcolo del valore **massimo fra tre numeri** presi in input, ovvero il numero maggiore.




Soluzione 3.1

Dati due numeri A e B verificare se A è il quadrato di B

- Prendiamo in input A e B utilizzando il parallelogramma.
- Dopo utilizzare il rombo per verificare se A è uguale al quadrato di B. Cioè poniamo come test: $A == B * B$.
- Se la condizione è vera visualizziamo semplicemente in output, utilizzando il parallelogramma, il messaggio *'A è il quadrato di B'*.
- Altrimenti se la condizione è falsa visualizziamo in output il messaggio: *'A non è il quadrato di B'*.





```
a = int(input('Inserisci il valore di a'))
b = int(input('Inserisci il valore di b'))
if a == b*b: # se il numero è negativo
    print(' a è il qua di b')
else:
    print(' a no è il qua di b')
```

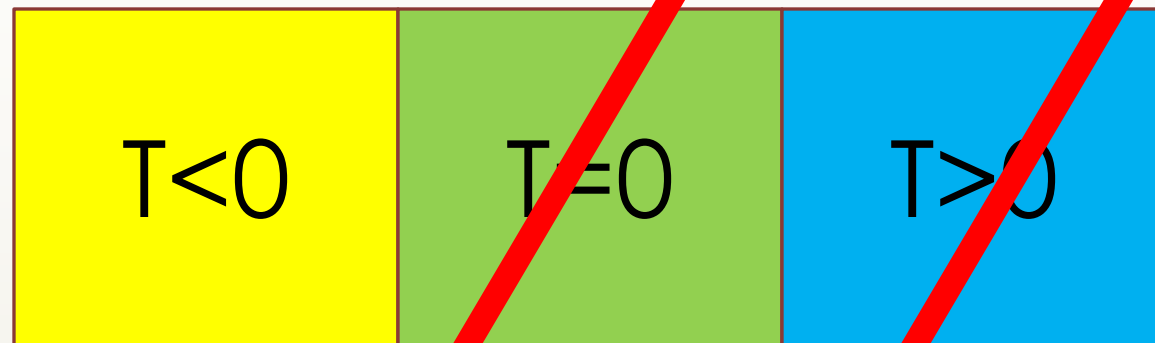


Soluzione n°3.2

Data una temperatura **T**, visualizzare se si tratta di una temperatura: **“sotto lo zero”**, **“uguale a zero”** o **“sopra lo zero”**.
Visualizzare il messaggio in output.

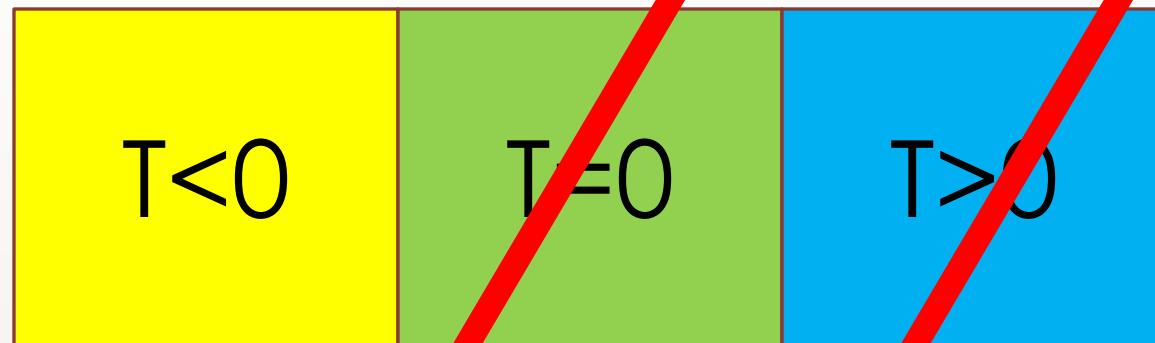
Soluzione n°3.2

➤ Possibili soluzioni

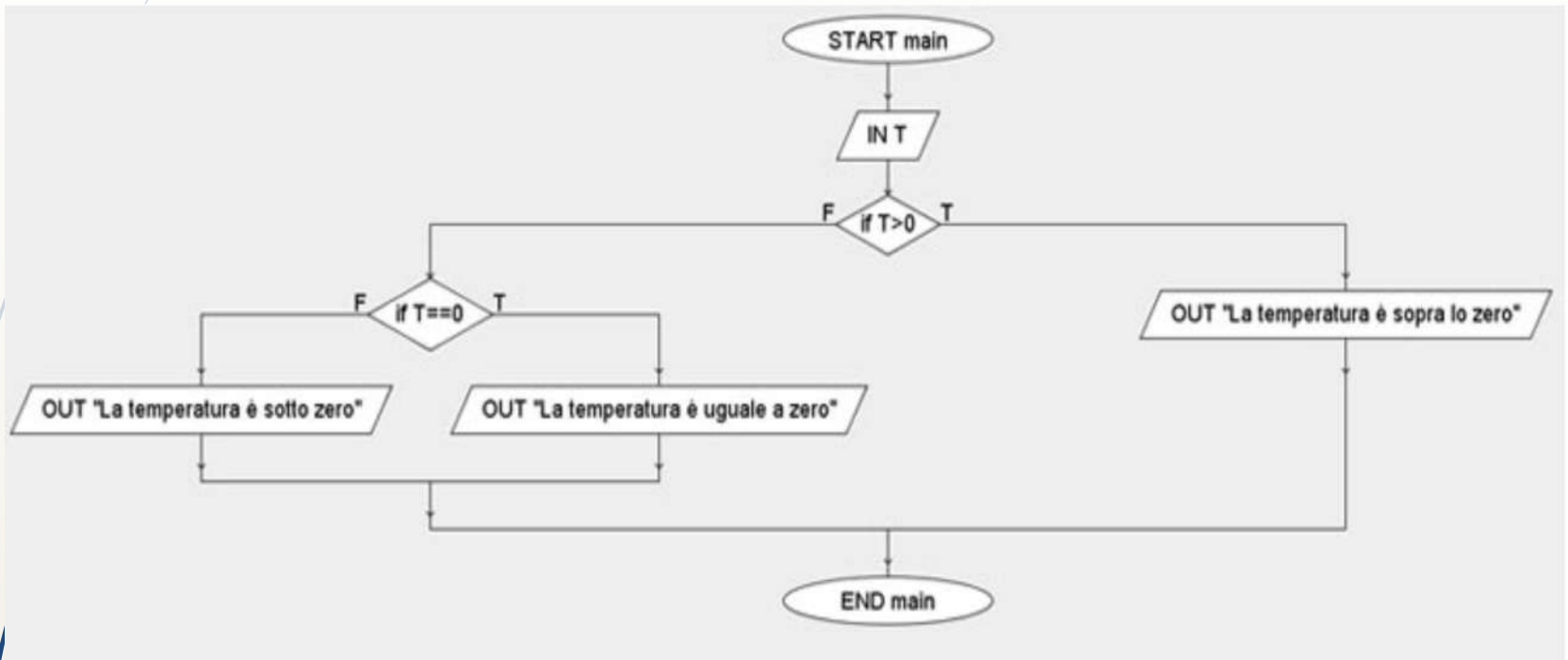


Soluzione n°3.2

➤ Possibili soluzioni



Soluzione n°3.2



Soluzione n°3.3

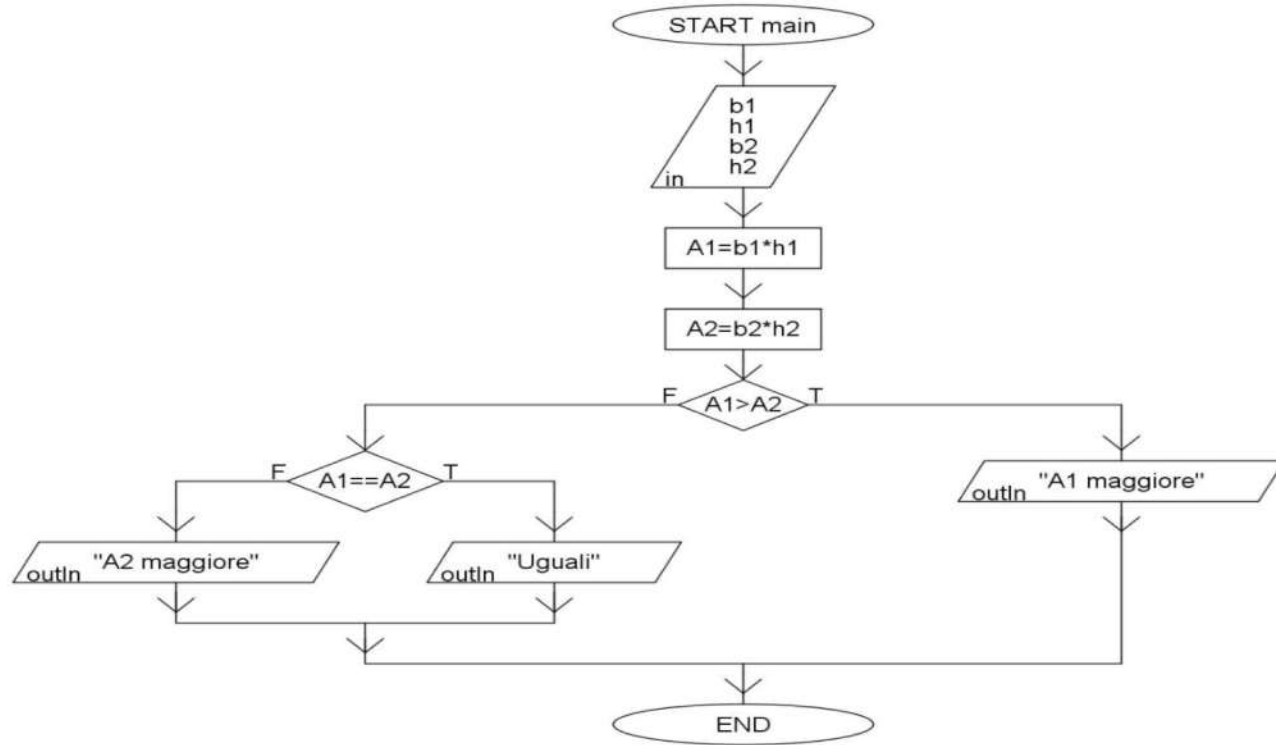
- Date le dimensioni di due rettangoli calcolarne l'area e determinare quale dei due ha la superficie maggiore.
- Per risolvere questo algoritmo dobbiamo prendere in input i dati necessari per calcolare l'area dei due rettangoli.
- Quindi prendiamo in input:
- **b1** – indica la base del primo rettangolo
- **h1** – indica l'altezza del primo rettangolo
- **b2** – indica la base del secondo rettangolo
- **h2** – indica l'altezza del secondo rettangolo
- Dopo calcoliamo l'area dei due rettangoli utilizzando due variabili A1 e A2.



Soluzione n°3.3

- Infine confrontiamo le due aree così ottenute per vedere quale delle due è maggiore. Quindi se $A1$ è maggiore di $A2$ scriviamo che $A1$ è maggiore. Altrimenti non possiamo ancora dire che $A2$ è maggiore di $A1$ in quanto dobbiamo verificare se sono uguali.
- Ecco quindi il diagramma di flusso che rappresenta l'algoritmo proposto.
- Notate che abbiamo messo le 4 variabili $b1$, $h1$, $b2$ e $h2$ per comodità nello stesso input.

Soluzione n°3.3



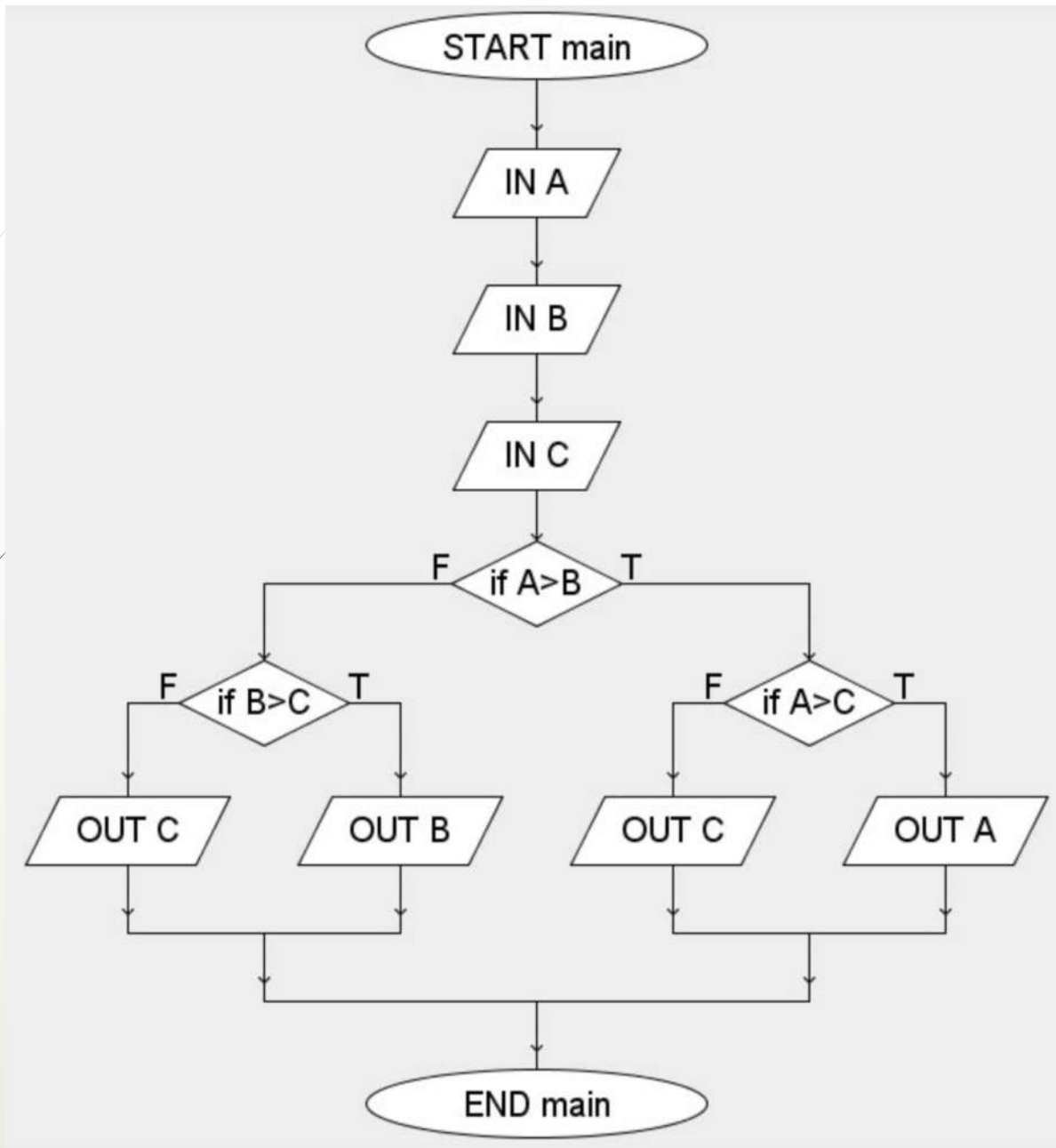


Soluzione n°3.4

Realizziamo un algoritmo per il calcolo del valore **massimo fra tre numeri** presi in input, ovvero il numero maggiore.

Soluzione n°3.4

- Innanzitutto prendiamo in input i tre numeri: A, B e C.
- Dopo effettuiamo il test **A>B** cioè ci chiediamo **A è maggiore di B?**
- Abbiamo allora due possibilità:
 1. Se il test è **vero** possiamo tralasciare B e confrontare A con C. Ci servirà dunque un altro rombo per effettuare il secondo test
 2. Se il test iniziale (**A>B**) è **falso** allora vuol dire che A è minore di B, quindi sicuramente A non sarà il maggiore, pertanto confronto B con C, in questo modo: **B>C**





Gli operatori Logici



- In questa soluzione utilizziamo l'operatore `||` che sta per **or**, cioè la funzione logica **O**.
- In questa soluzione utilizziamo l'operatore `&&` che sta per **and**, cioè la funzione logica **E**.



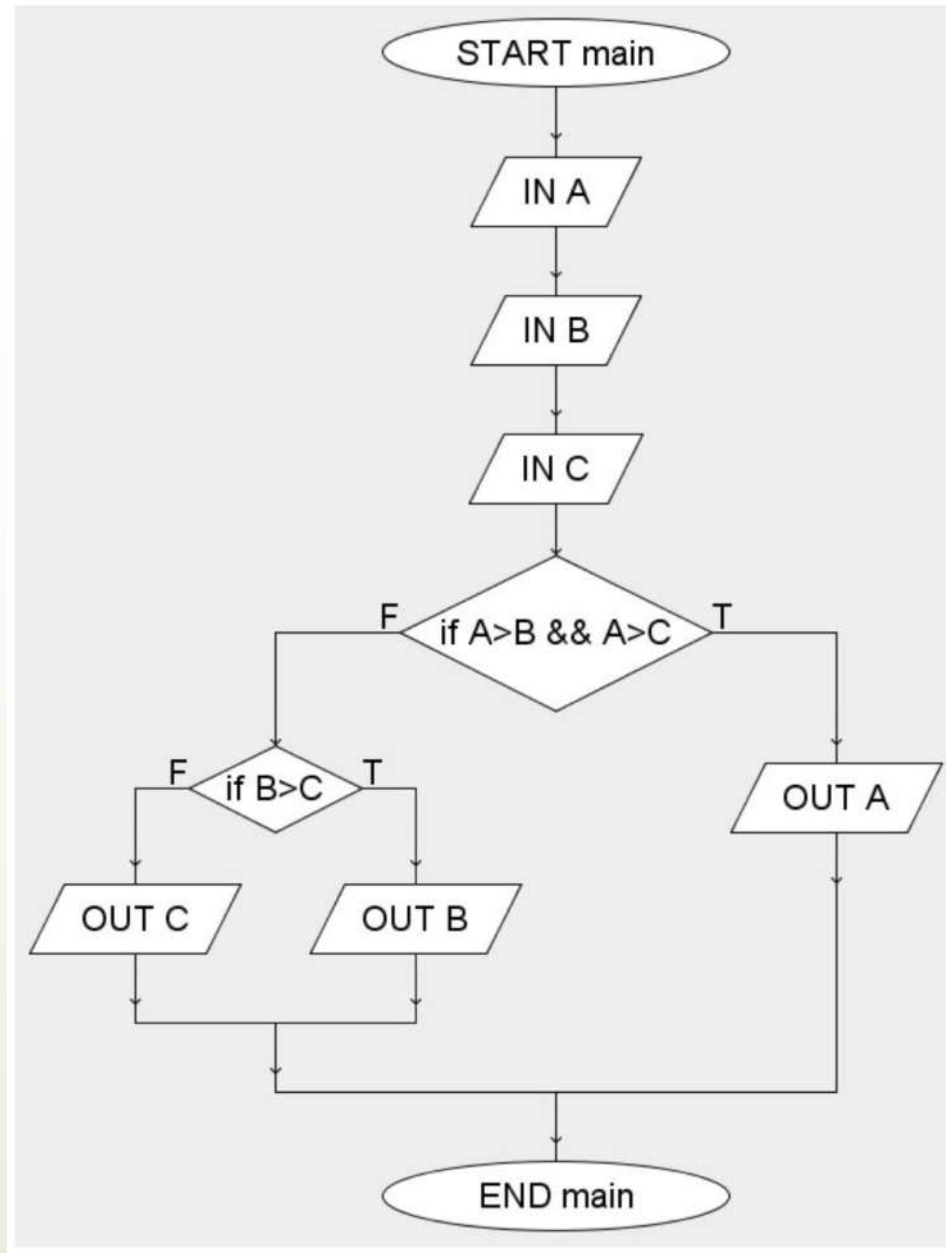
Soluzione con gli operatori logici

- L'algoritmo si può risolvere anche in un altro modo. Ad esempio utilizzando gli operatori logici.
- In questa soluzione utilizziamo l'operatore **&&** che sta per **and**, cioè la funzione logica **E**.
- Poniamo come prima condizione che A sia maggiore di B e contemporaneamente che A sia maggiore di C. Dunque in questo modo: $A > B \ \&\& \ A > C$.



Soluzione con gli operatori logici

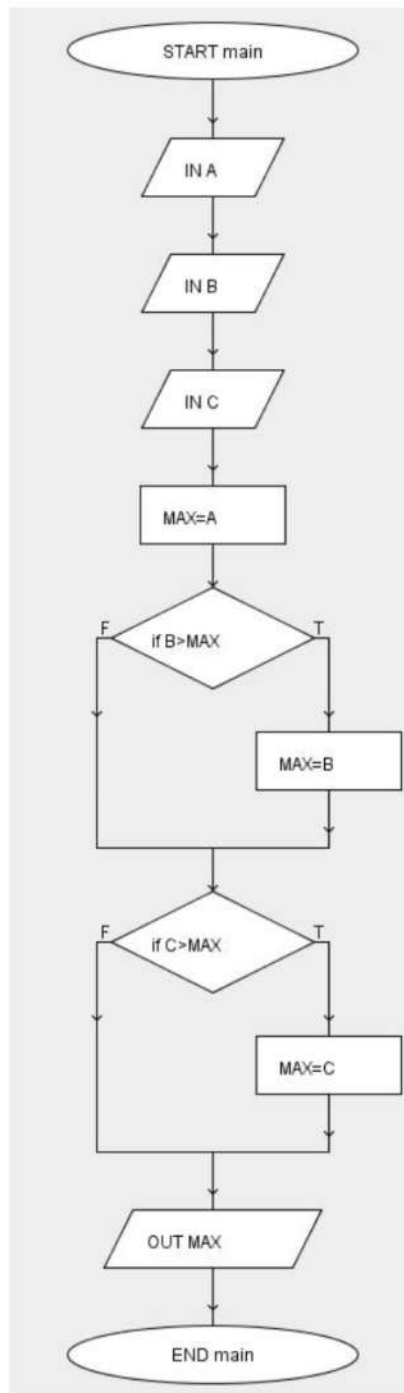
- Se tale condizione è vera chiaramente A è il maggiore.
- Altrimenti se $A > B \ \&\& \ A > C$ è falsa, vuol dire che A non può essere il maggiore, perché chiaramente non è comunque maggiore di entrambi. Dunque possiamo escludere A dal prossimo confronto e valutare solo B e C.
- Quindi controlliamo solo se B è maggiore di C e se tale condizione è vera il massimo sarà B, altrimenti il massimo sarà C.

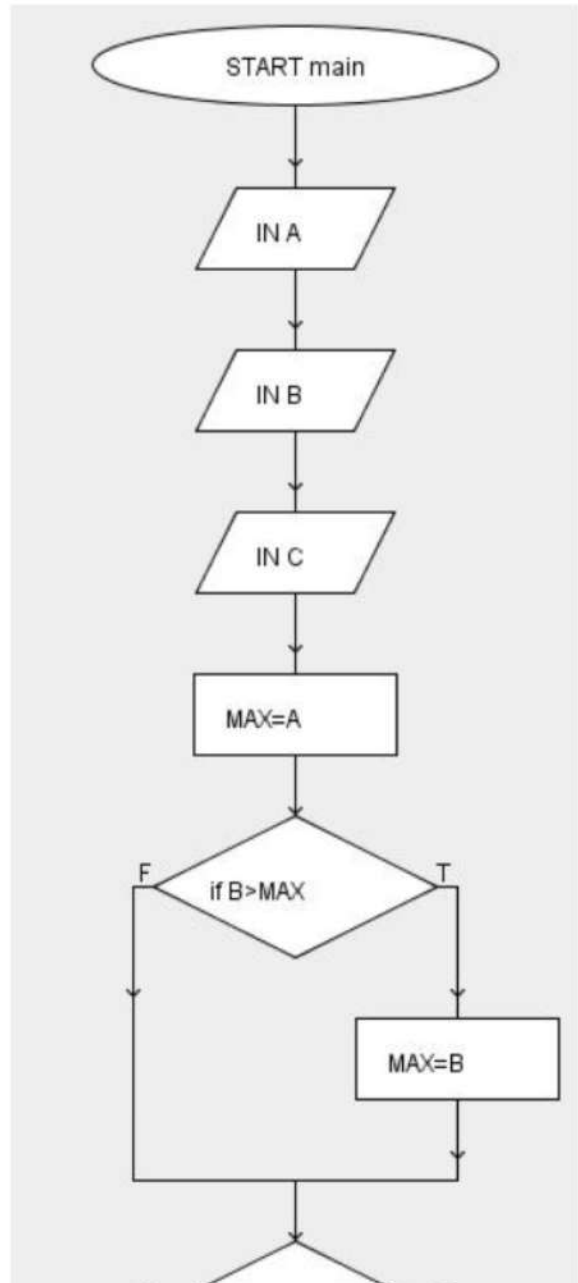


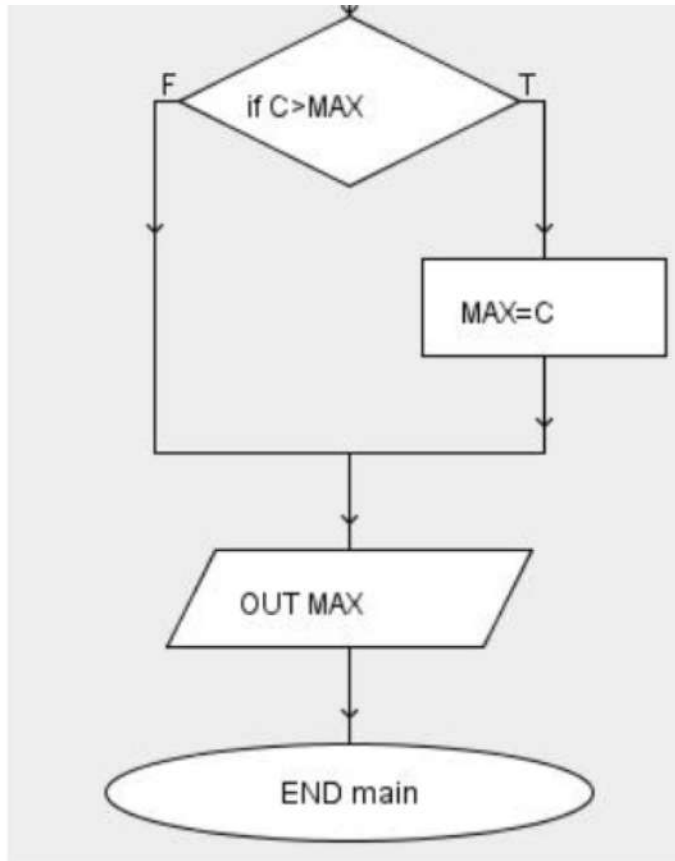


Soluzione con una variabile di comodo

- Utilizziamo un variabile di comodo **MAX** dove memorizziamo il primo valore preso in input A.
- se **B è maggiore di MAX** sostituisco il valore, altrimenti non faccio nulla in quanto MAX è più grande.
- Dopo controllo se **C è maggiore di MAX** e se vero sostituisco il valore, altrimenti come prima vuol dire che MAX è il maggiore.







Soluzione in Python

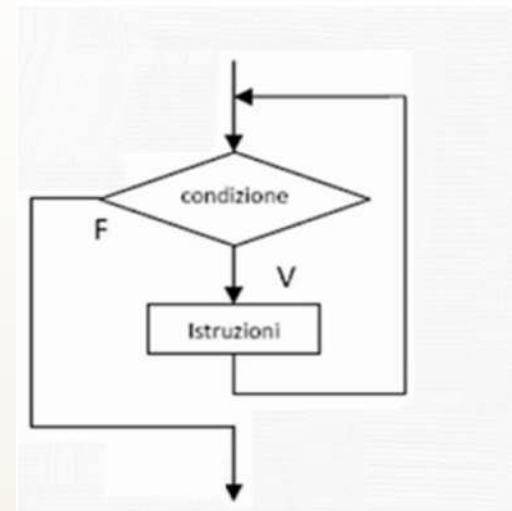
```
a = int(input('Inserisci il valore di a'))
b = int(input('Inserisci il valore di b'))
c = int(input('Inserisci il valore di c'))
max=a
if b>max: # se b è maggiore
    max=b
if c>max: # se b è maggiore
    max=c
print('Il valore max è = ', max)
```

Iterazione



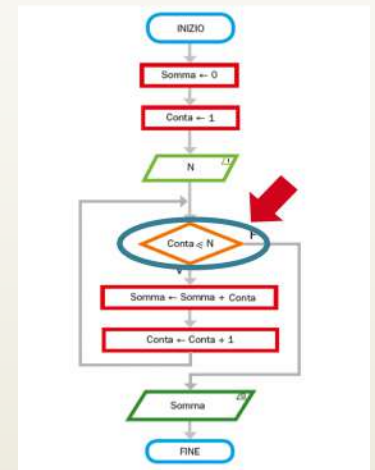
Iterazione

- In un **algoritmo** può capitare che alcune operazioni debbano essere eseguita più di una volta, cioè **ripetute in modo identico**
- La ripetizione di un insieme di istruzioni prende il nome di **iterazione**
- Il gruppo di istruzioni ripetute prende il nome di **corpo del ciclo**

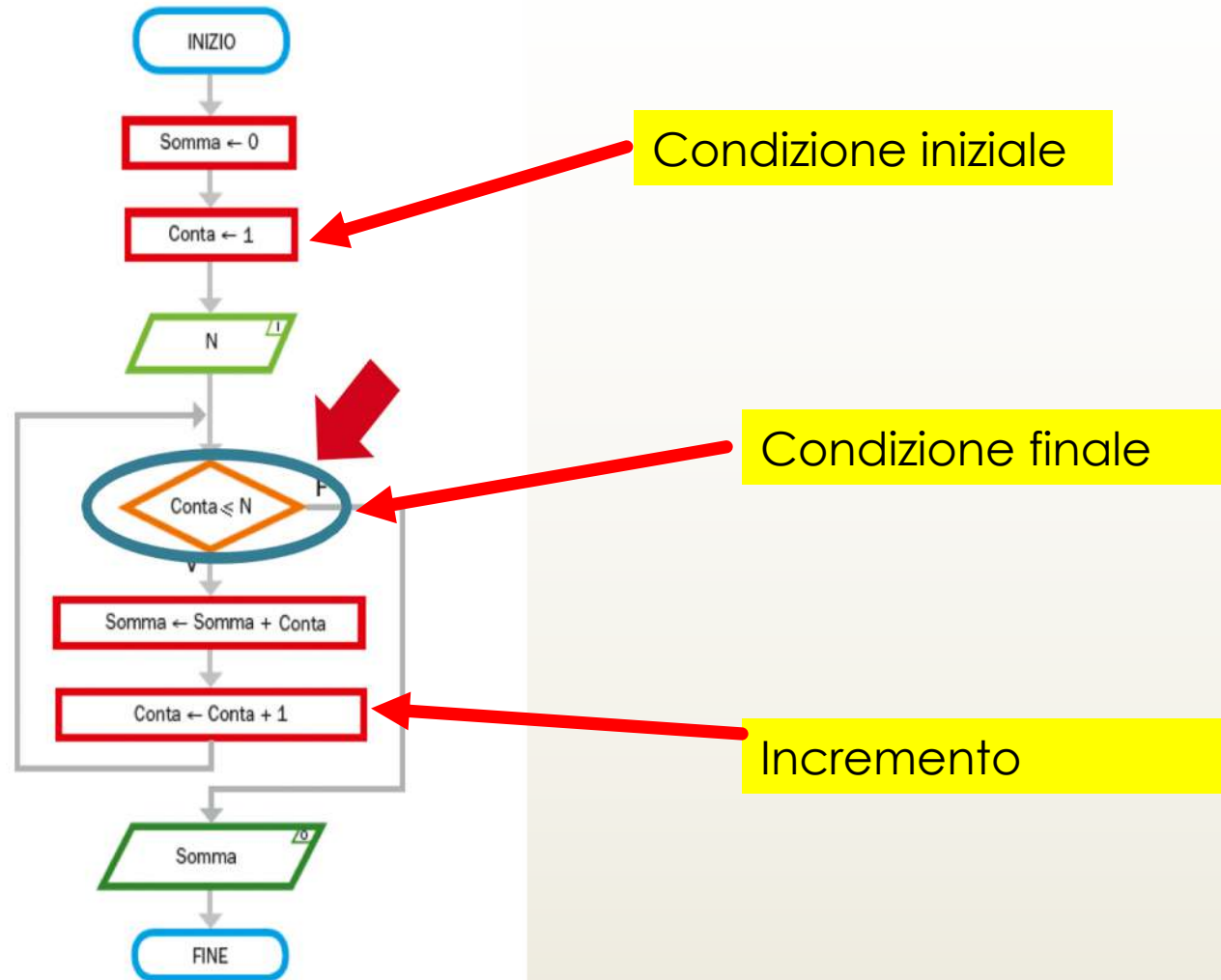


Componenti essenziali

- Ad ogni iterazione dobbiamo sempre impostare le seguenti indicazioni:
 1. Condizione iniziale, il valore da cui partire
 2. Condizione finale, il valore da cui uscire
 3. Incremento, quanto deve incremento



Componenti essenziali



Istruzioni iterazione

Iterazione può essere tradotto in un linguaggio di programmazione può essere:

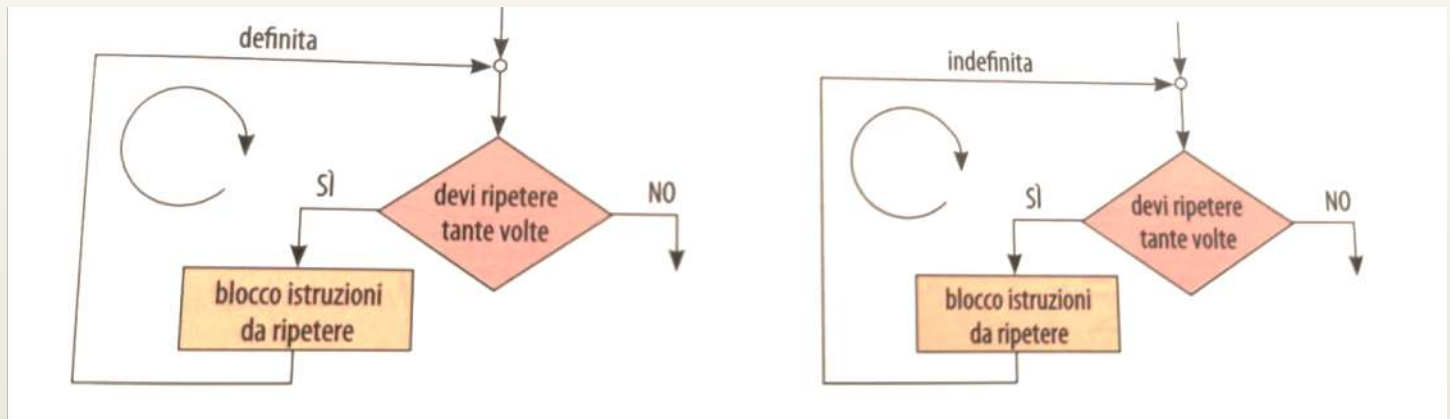
► While

```
i=1
while i<10:
    print(i)
    i=i+1
```

► For

```
i=1
for i in range(1,10):
    print(i)
```

1
2
3
4
5
6
7
8
9



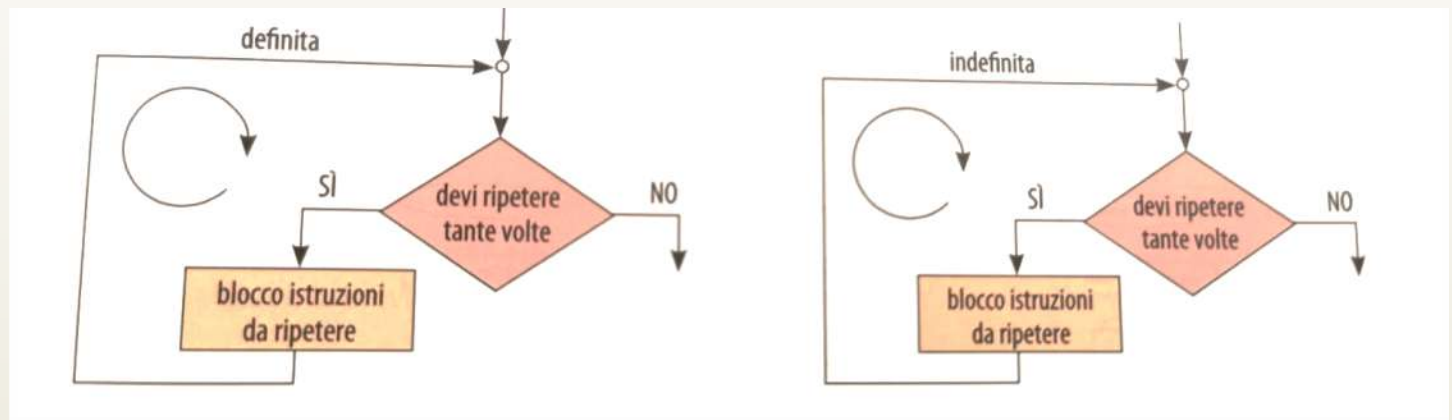
Classificazione dell'Iterazione

► Iterazione Definita

- quando il numero di iterazione è noto

► Iterazione Indefinita

- quando il numero di iterazione è sconosciuto



Esempi

ITERAZIONI DEFINITE

Fai 10 giri di corsa.
Ascolta 5 canzoni.
Per 20 volte scrivi "devo fidarmi di più".
Invia 30 sms ai tuoi amici.
Imbianca con 3 mani la tua stanza.
Tra 210 giorni è finita la scuola.

ITERAZIONI INDEFINITE

Mescola la pasta finché è cotta.
Innaffia le piante finché basta.
Mangia finché sei sazio.
Mentre piove fatti una dormita.
Mentre non sei stanco, studia.
Dormi finché hai sonno.

Classificazione dell'Iterazione

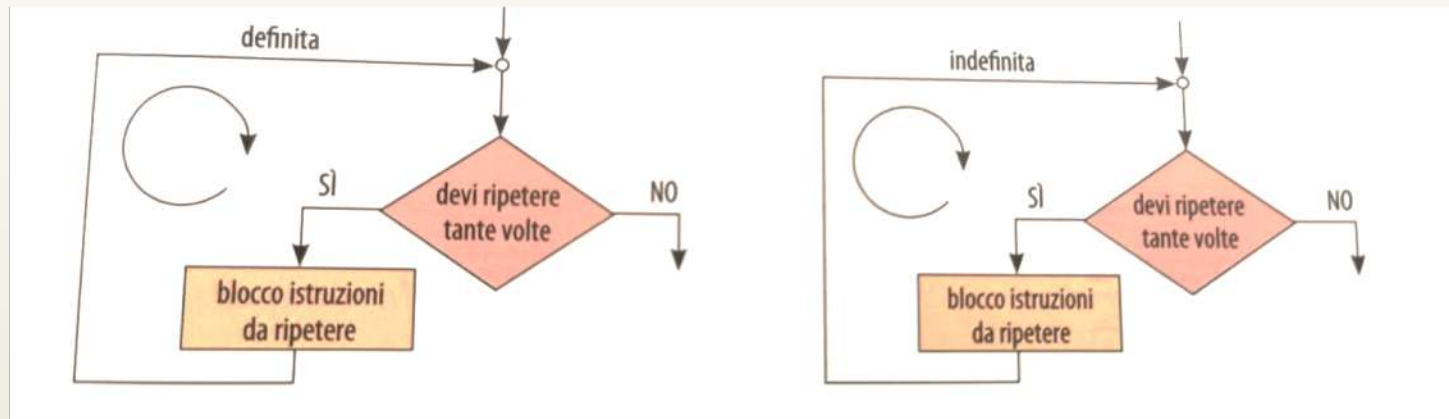
Possiamo osservare che i due scemi sono identici, cambia solo il controllo/test di uscita dal ciclo:

► Iterazione definita

- Il controllo/test viene fatto sul numero del contatore

► Iterazione Indefinita

- Il controllo/test viene fatto nel ciclo ed è una qualunque condizione logica.



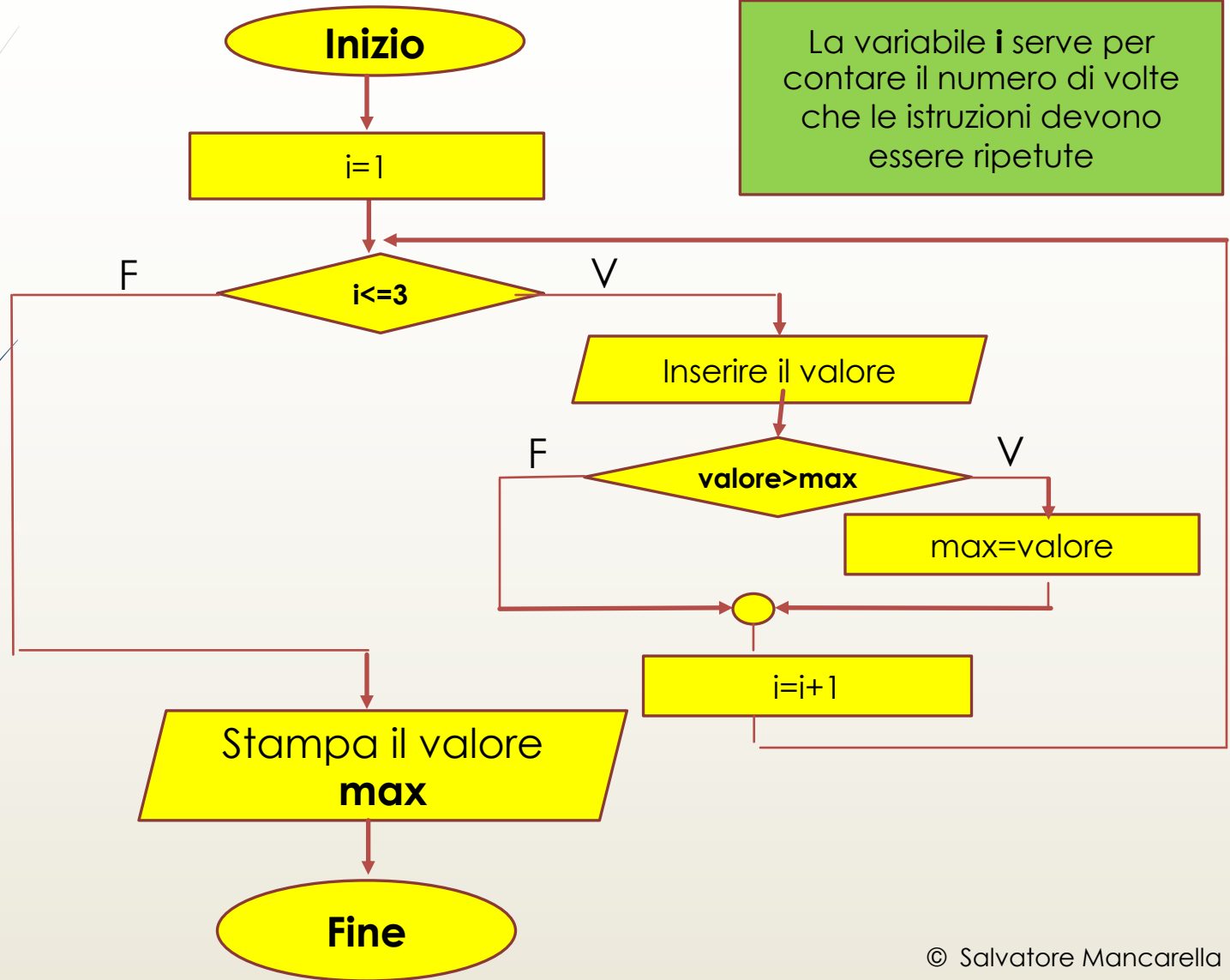


Esercitazioni n°4 (iterazione definita)

1. Realizziamo un algoritmo per il calcolo del valore **massimo fra tre numeri** presi in input, ovvero il numero maggiore.
2. Calcolare la somma, il prodotto e la media di 10 valori inseriti dall'utente.
3. Calcolare il valore minimo di 10 valori inseriti dall'utente.
4. Realizza un programma che determini la potenza intera di un numero utilizzando soltanto l'operatore prodotto. Il valore della potenza (l'esponente) e il numero (la base) sono immessi dall'utente. Utilizzare il ciclo while

Soluzione Es. n°4.1

La variabile i serve per contare il numero di volte che le istruzioni devono essere ripetute



Soluzione in Python

```
a = int(input('Inserisci il valore di a'))
b = int(input('Inserisci il valore di b'))
c = int(input('Inserisci il valore di c'))
max=a
if b>max: # se b è maggiore
    max=b
if c>max: # se b è maggiore
    max=c
print('Il valore max è = ', max)
```

```
max=0
for i in range(1,3):
    valore = int(input('Inserisci il valore '))
    if valore>max: # se b è maggiore
        max=valore
print('Il valore max è = ', max)
```

Soluzione n°4.2

```
#Calcolare la somma il prodotto e la media di 10 valori inseriti dall'utente
somma=0
prodotto=1
media=0
i=1
while i<=3:
    valore = int(input('Inserisci il valore '))
    somma = somma + valore
    prodotto = prodotto*valore
    i=i+1
media = somma/10
print('La somma è = ', somma)
print('Il prodotto è = ', prodotto)
print('La media è = ', media)
```

Esercitazioni n°4

(iterazione indefinita)

5. Un utente vuole acquistare una macchina, ha come budget a disposizione 800€. Il programma deve segnalare un messaggio se l'utente può permettersi di fare l'acquisto.
6. Calcolare la somma, il prodotto e la media di n valori immessi dall'utente, dove non si conosce in partenza il numero preciso.
7. Si vuole costruire un gazebo dopo aver inserito il budget di spesa massima, si determina il totale del preventivo inserendo il prezzo dei vari componenti, inviando un messaggio appena si supera il Budget a disposizione.
8. Si riceve come dato d'ingresso una sequenza di numeri terminante per 0, i numeri sono al massimo 100, non è conosciuta a priori la lunghezza della sequenza. Visualizza il valore del numero maggiore e di quello minore

Soluzione n°4.5(For) (iterazione indefinita)

Condizione iniziale

Condizione finale

```
i=1
for i in range(1,100):
    valore = int(input('Inserisci il valore della macchina'))
    if valore>800:
        print('Il valore della macchina è fuori del tuo budget di 800€')
        break
print('Puoi acquistare al macchina perchè inferiore al tuo budget di 800€')
```

Incremento → con
l'istruzione for
automaticamente
incrementa di 1

Soluzione n°4.5 (While) (iterazione indefinita)

Condizione iniziale

Condizione finale

Incremento

```
i=1
while i<100:
    valore = int(input('Inserisci il valore della macchina'))
    if valore>800:
        print('Il valore della macchina è fuori del tuo budget di 800€')
        break
    i=i+1
    print('Puoi acquistare al macchina perchè inferiore al tuo budget di 800€')
```

Soluzione n°4.6

```
#Calcolare la somma il prodotto e la media di n valori inseriti dall'utente
somma=0
prodotto=1
media=0
i=1
while i<=100:
    valore = int(input('Inserisci il valore '))
    if valore == 0:
        break
    somma = somma + valore
    prodotto = prodotto*valore
    i=i+1

media = somma/(i-1)
print('La somma è = ', somma)
print('Il prodotto è = ', prodotto)
print('La media è = ', media)

...
```

Soluzione m°4.7

```
#Calcolare la somma il prodotto e la media di 10 valori inseriti dall'utente

budget = int(input('Inserire in budget iniziale '))
somma=0
i=1
while i<=100:
    valore = int(input('Inserisci il valore '))
    somma = somma + valore
    if valore == 0:
        break
    if somma > budget:
        break
    i=i+1
print('Il totale del preventivo è = ', somma)
```

Esercitazioni n°4

(riepilogo iterazione e sequenza)

9. Sviluppare l'algoritmo che calcoli la successione di Fibonacci
10. Dati in input i valori di due variabili **a** e **b**, vogliamo scambiare i valori, così che al termine **a** contiene il valore di **b** e **b** quello di **a**.
11. Si vuole costruire vari bilocali, ognuno è composto da una stanza da letto un bagno e una cucina, inserire le rispettive misure e calcolare l'area totale, considerando che dalle regole urbanistiche è ammessa la realizzazione al massimo di 60mq, non è conosciuta a priori il numero di tentativi che l'utente deve fare. Il programma termina quando l'area totale è maggior a quella ammessa, visualizzando il relativo messaggio di non realizzabilità altrimenti conferma l'ammissibilità alla realizzazione.
12. Calcolare la somma, il prodotto e la media di n valori immessi dall'utente, dove non si conosce in partenza il numero preciso, bisogna considerare solo i numeri pari.
13. Si riceve come dato d'ingresso una sequenza di numeri terminante per 0, i numeri sono al massimo 100, non è conosciuta a priori la lunghezza della sequenza. Visualizza il valore del numero maggiore e di quello minore dei valori multipli di 3.
14. Si riceve come dato d'ingresso una sequenza di numeri terminante per 0, i numeri sono al massimo 100, non è conosciuta a priori la lunghezza della sequenza. Visualizza il valore del numero maggiore e di quello minore dei valori pari.

Soluzione n°4.9

- Ricordiamo che la successione di Fibonacci è una successione di numeri interi positivi in cui ciascun numero a cominciare dal terzo è la somma dei due precedenti eccetto i primi due che sono 1, 1

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, ...

$$1 + 1 = 2$$

$$1 + 2 = 3$$

$$2 + 3 = 5$$

$$3 + 5 = 8$$

$$5 + 8 = 13$$

$$8 + 13 = 21$$

$$13 + 21 = 34$$

$$21 + 34 = 55$$

$$34 + 55 = 89$$

$$55 + 89 = 144$$

$$89 + 144 = 233$$

$$144 + 233 = 377$$

$$233 + 377 = 610$$

$$377 + 610 = 987$$

Soluzione n°4.9

```
#Calcola la successione di fibonacci
```

```
N=int(input('Quanti numeri?: ' ))
```

```
a,b=1,1
```

```
print(a)
```

```
print(b)
```

```
for i in range(N):
```

```
    c=a+b
```

```
    a=b
```

```
    b=c
```

```
    print(c, end=' ')
```

Soluzione 4.10

Dati in input i valori di due variabili **a** e **b**, vogliamo scambiare i valori, così che al termine **a** contiene il valore di **b** e **b** quello di **a**.

- ▶ Ecco dunque un esempio:
 - ▶ Se $a=5$ e $b=4$ vogliamo che al termine del nostro algoritmo sia $a=4$ e $b=5$.
 - ▶ Ragioniamo quindi su una possibile soluzione! Sicuramente, per scambiare questi valori, una possibile soluzione può essere quella di utilizzare una variabile temporanea di appoggio.
 - ▶ Chiamo questa terza variabile **temp**, e procedo in questo modo:

Soluzione n°10

```
# Scambio del valore di due variabili
a=int(input('Inserisci a:'))
b=int(input('Inserisci b:'))
print('I valori inseriti sono a:', a, ' e b: ', b)
print('Adesso scambio i valori')
temp=a
a=b
b=temp
print('I valori scambiati sono a:', a, ' e b: ', b)
```



Soluzione alternativa (assegnazione multipla)

- Introduciamo perciò il concetto di **assegnazione multipla**.
- Con l'**assegnazione multipla** in python si possono assegnare più variabili alla volta.
- Quindi ad esempio:
 - **a=b=5**
 - In questo caso sia a che b assumono il valore di 5.
 - Oppure un'altra assegnazione che posso fare è questa:
 - **a, b= 5, 4**
 - In questo modo assegno ad **a** il valore 5 e a **b** il valore 4.

Soluzione alternativa (assegnazione multipla)

Riprendiamo gli Esercizio n°4.9 e 4.10 e proviamo a trovare la soluzione utilizzando il concetto di **assegnazione multipla**.

► Soluzione 4.9

```
#Calcola la successione di fibonacci

N=int(input('Quanti numeri?: ' ))
a,b=1,1
print(a)
print(b)

for i in range(N):
    c=a+b
    a=b
    b=c
    print(c, end=' ')
```

```
#Calcola la successione di fibonacci
#con assegnazione multipla

N=int(input('Quanti numeri?: ' ))
a,b=1,1
for i in range (N):
    print(a, end=' ')
    a,b=b,a+b
    print()
```

Soluzione n°4.10

```
# Scambio del valore di due variabili
a=int(input('Inserisci a:'))
b=int(input('Inserisci b:'))
print('I valori inseriti sono a:', a, ' e b: ', b)
print('Adesso scambio i valori')
temp=a
a=b
b=temp
print('I valori scambiati sono a:', a, ' e b: ', b)
```

```
# Scambio del valore di due variabili
#con assegnazione multipla
a=int(input('Inserisci a:'))
b=int(input('Inserisci b:'))
print('I valori inseriti sono a:', a, ' e b: ', b)
print('Adesso scambio i valori')
a,b=b,a
print('I valori scambiati sono a:', a, ' e b: ', b)
```

Soluzione n°4.12

➤ Esercitazione n°4.12

```
# Calcolare la somma il prodotto e la media di n valori inseriti dall'utente
# considerando solo i numeri pari
somma=0
prodotto=1
media=0
conta_pari=0
i=1
while i<=100:
    valore = int(input('Inserisci il valore '))
    if valore == 0:
        break
    if(valore%2==0):
        somma = somma + valore
        prodotto = prodotto*valore
        conta_pari=conta_pari+1
    i=i+1
media = somma/conta_pari
print('La somma è = ', somma)
print('Il prodotto è = ', prodotto)
print('La media è = ', media)
```

Soluzione n°4.13

➤ Esercitazione n°4.13

```
# Calcolare il minimo e il massimo di n valori inseriti dall'utente
# solo dei valori multipli di 3
max=0
primo_valore=0
i=1
while i<=100:
    valore = int(input('Inserisci il valore '))
    if valore == 0:
        break
    if(valore%3==0):
        if primo_valore == 0:
            primo_valore=1
            min=valore
        if primo_valore==1 and valore<min:
            min=valore
        if valore > max:
            max = valore
    i=i+1
print('Il valore minimo è uguale a ', min)
print('Il valore massimo è uguale a ', max)
```

I vettori



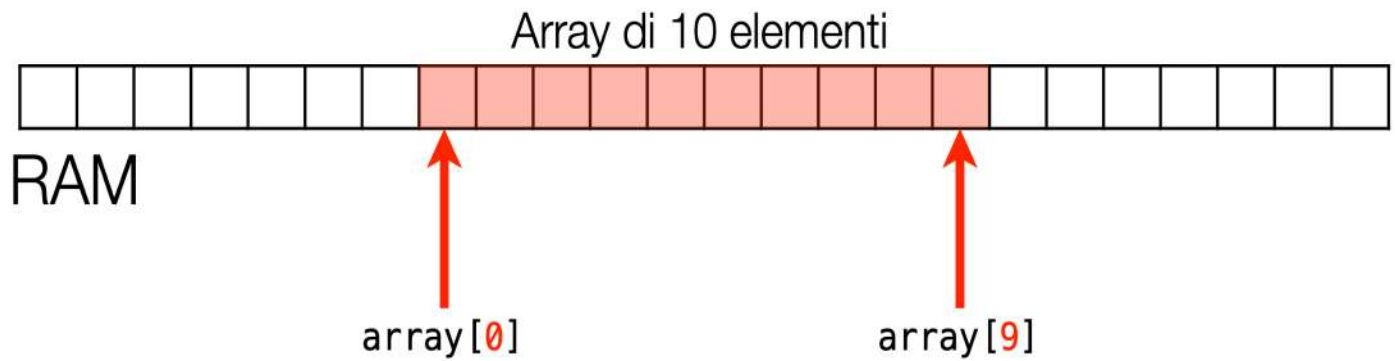
Differenza tra una variabile e un vettore

Variabile
Int **a**

2

Vettore «**vet**»

Posizione 0	2	vet[0]
Posizione 1	4	vet[1]
Posizione 2	6	vet[2]
Posizione 3	9	vet[3]



Dichiarazione di un vettore in Python

- Per inizializzare un vettore nel codice

```
n=3
numeri=[2,3,5]
print('I numeri inserite nel vettore/lista sono: ')
for i in range(n):
    print(numeri[i])
```

Inserimento i valori nel vettore

visualizza i valori nel vettore

- I dati sono inseriti dall'utente

```
n=3
numeri=[]
for i in range(n):
    numero=int(input('Inserisci un numero: '))
    numeri.append(numero)
print('I numeri inseriti nel vettore/lista sono: ')
for i in range(n):
    print(numeri[i])
```

Inserimento i valori nel vettore

visualizza i valori nel vettore



Esercitazioni n°5 (vettori)

1. Realizziamo un algoritmo per il calcolo del valore **massimo fra tre numeri** presi in input, ovvero il numero maggiore.
2. Popolare una lista con 20 numeri a piacere. Dopo l'inserimento visualizzare tutti gli elementi della lista con il relativo indice. Dopo trovare il valore massimo tra quelli inseriti nella lista.
3. L'utente dopo aver inserito 10 valori in una lista. Calcolare la somma, il prodotto e la media di 10 valori inseriti dall'utente.
4. L'utente dopo aver inserito 10 valori in una lista. Calcolare il valore minimo di 10 valori inseriti dall'utente.
5. L'utente dopo aver inserito 5 valori in un vettore. Visualizzare solo i numeri pari e la loro somma e il loro prodotto.
6. L'utente dopo inserito 5 valori in un vettore. Deve eliminare solo i numeri pari.
7. In una lista, di nome casuali, inserire n numeri interi random tra 1 e 30. Dopo l'inserimento visualizzare gli elementi della lista con gli indici. Poi contare e visualizzare quanti numeri non multipli di 3 sono stati inseriti nella lista.
8. Realizza un programma che determini la potenza intera di un numero utilizzando soltanto l'operatore prodotto. Il valore della potenza (l'esponente) e il numero (la base) sono immessi dall'utente. Utilizzare il ciclo while
9. L'utente dopo inserito 5 valori in un vettore. Deve eliminare solo i numeri dispari

Soluzione n°5.1

- Con dati inseriti dall'utente

```
n=3
numeri=[]
for i in range(n):
    numero=int(input('Inserisci un numero: '))
    numeri.append(numero)
print('I numeri inseriti nel vettore/lista sono: ')
for i in range(n):
    print(numeri[i])
massimo=numeri[0]
for i in range(n):
    if numeri[i]>massimo:
        massimo=numeri[i]
print('Il valore maggiore inserito è: ', massimo)
```

Soluzione n°5.1

➤ Con dati inseriti nel codice

Codice

```
n=3
numeri=[2,3,5]
print('I numeri inserite nel vettore/lista sono: ')
for i in range(n):
    print(numeri[i])
massimo=numeri[0]
for i in range(n):
    if numeri[i]>massimo:
        massimo=numeri[i]
print('Il valore maggiore inserito è: ', massimo)
```

Risultato

```
I numeri inserite nel vettore/lista sono:
2
3
5
Il valore maggiore inserito è: 5
```

Soluzione n°5.5

```
n=5
somma=0
media=0
prodotto=1
conta_pari=0;
vet=[]
for i in range(n):
    numero=int(input('Inserisci elemento: '))
    vet.append(numero)

for i in range(n):
    if vet[i]%2==0:
        print('Elemento pari sono nella posizione:', i, 'vale', vet[i])

for i in range(n-1,-1,-1):
    if vet[i]%2==0:
        somma = somma+vet[i]
        prodotto =prodotto*vet[i]
        conta_pari= conta_pari+1
media=somma/conta_pari
print('La somma dei valori pari è :',somma)
print('La media dei valori pari è :',media)
print('La prodotto dei valori pari è :',prodotto)
```



Soluzione n°5.6

L'utente dopo inserito 5 valori in un vettore. Deve eliminare solo i numeri pari.

- Per eliminare gli elementi occorre partire dalla coda della lista, pertanto invertiamo l'intervallo del range in modo da considerare gli elementi che vanno da $n-1$ a 0 .
- Non possiamo iniziare dall'inizio perché, nel momento in cui elimino il primo elemento, in automatico il secondo elemento passa in prima posizione al termine della prima iterazione. Quindi alla seconda iterazione avremo un elemento in meno e si avrà l'errore '**out of range**'.
- Dobbiamo personalizzare la funzione **range**

Soluzione n°5.6

- Come personalizzare la funzione range



- Per eliminare un valore nel vettore utilizziamo la funzione **remove**

```
for i in range(n-1,-1,-1):  
    if vet[i]%2==0:  
        vet.remove(vet[i])
```

Soluzione n°5.6

➤ Il codice Python

```
n=5
vet=[]
for i in range(n):
    numero=int(input('Inserisci elemento: '))
    vet.append(numero)

for i in range(n):
    print('Elemento in posizione:', i, 'vale', vet[i])

for i in range(n-1,-1,-1):
    if vet[i]%2==0:
        print('Elemento eliminato nella posizione', i, ' elemento', vet[i])
        vet.remove(vet[i])
print('Il nuovo vettore ha i seguenti elementi!')
for i in range(len(vet)):
    print('Nella posizione', i, ' elemento', vet[i])
```

Soluzione n°5.6

➤ Risultato ottenuto

```
Inserisci elemento: 3
Inserisci elemento: 4
Inserisci elemento: 6
Inserisci elemento: 7
Inserisci elemento: 9
Elemento in posizione: 0 vale 3
Elemento in posizione: 1 vale 4
Elemento in posizione: 2 vale 6
Elemento in posizione: 3 vale 7
Elemento in posizione: 4 vale 9
Elemento eliminato nella posizione 2 elemento 6
Elemento eliminato nella posizione 1 elemento 4
Il nuovo vettore ha i seguenti elementi!
Nella posizione 0 elemento 3
Nella posizione 1 elemento 7
Nella posizione 2 elemento 9
```

Soluzione n°5.7

Per generare i numeri interi **random** da 1 a 30 abbiamo bisogno di importare il modulo random.

- Possiamo importare tutto il modulo:

- `import random`

- E quindi dopo utilizzare:

- `numero=random.randint(1,30)`

- Oppure possiamo importare solo la funzione che ci serve in questo modo:

- `from random import randint`

- E poi utilizzare semplicemente:

- `randint(1,30)`

Soluzione n°5.7

- Quindi dopo aver importato il modulo random,

```
import random
```

- Chiediamo di inserire la quantità di numeri random che si vuole inserire nella lista, cioè **n**.

```
numero=random.randint(1,30)
```

- Poi inizializziamo casuali alla lista vuota.
- Popoliamo la lista con i numeri random da 1 a 30 e visualizziamo in output gli elementi con i relativi indici.
- Infine controlliamo quanti e quali elementi che non sono multipli di 3 sono stati inseriti.

Soluzione n°5.7

```
import random
n=int(input('Quantità di numeri: '))
casuali=[]
c=0

for i in range(n):
    numero=random.randint(1,30)
    casuali.append(numero)

print('I numeri casuali inseriti nella lista sono: ')

for i in range(n):
    print(casuali[i],end=' ')
print()
print('I numeri inseriti che non sono multipli di 3: ')

m=0
for i in range(n):
    if casuali[i]%3!=0:
        print(casuali[i], end=' ')
        m+=1
print('I numeri che non sono multipli di 3 in tutto sono: ', m)
```

I cicli annidati



Esercitazioni n°6 (cicli annidati)

1. Realizziamo un programma che visualizzi in output un **rettangolo di asterischi**, prendendo in input la base e l'altezza.
2. Realizziamo un programma che visualizzi in output un **quadrato di asterischi**, prendendo in input la base e l'altezza.
3. Realizziamo in questa lezione una piramide di asterischi in python, come quella mostrata nella figura sotto, avente come altezza 6 asterischi.

```
      *
     ***
    *****
   *********
  ***********
 *****
*****
```

Soluzione n°6.1

Realizziamo un programma che visualizzi in output un **rettangolo di asterischi**, prendendo in input la base e l'altezza.

- Ipotizziamo la
 - base = 4
 - altezza = 5
- **i** si riferisce al conteggio delle righe
- **J** si riferisce al conteggio delle colonne

```
b=int(input('Base: '))
h=int(input('Altezza: '))
for i in range(h): # ciclo per le righe
    for j in range(b): # ciclo per le colonne
        print('* ', end=' ')
    print()
```

```
Base: 4
Altezza: 5
* * * *
* * * *
* * * *
* * * *
* * * *
```

Soluzione n°6.1

➤ Passo 1

➤ $i=0$ per le righe

➤ entra nel secondo con $j=0$ e ripete l'istruzione

```
print('* ', end=' ')
```

➤ per 4 volte in quanto abbiamo inserito come base 4

➤ Ottenendo la stampa degli asterischi per la prima riga

```
* * * *
```

```
b=int(input('Base: '))
h=int(input('Altezza: '))
for i in range(h): # ciclo per le righe
    for j in range(b): # ciclo per le colonne
        print('* ', end=' ')
    print()
```

Soluzione n°6.1

➤ Passo 2

➤ i=1 per le righe

➤ entra nel secondo con j=0 e ripete l'istruzione

```
print('* ', end=' ')
```

➤ per 4 volte in quanto abbiamo inserito come base 4

➤ aggiungendo alla prima la seconda riga di asterischi ottenendo

```
b=int(input('Base: '))
h=int(input('Altezza: '))
for i in range(h): # ciclo per le righe
    for j in range(b): # ciclo per le colonne
        print('* ', end=' ')
    print()
```

```
* * * *
* * * *
```

Soluzione n°6.1

➤ Passo 3

➤ i=2 per le righe

➤ entra nel secondo con j=0 e ripete l'istruzione

```
print('* ', end=' ')
```

➤ per 4 volte in quanto abbiamo inserito come base 4

➤ aggiungendo alla seconda la terza riga di asterischi ottenendo

```
b=int(input('Base: '))
h=int(input('Altezza: '))
for i in range(h): # ciclo per le righe
    for j in range(b): # ciclo per le colonne
        print('* ', end=' ')
    print()
```

```
* * * *
* * * *
* * * *
```

Soluzione n°6.1

➤ Passo 4

➤ i=3 per le righe

➤ entra nel secondo con j=0 e ripete l'istruzione

```
print('* ', end=' ')
```

➤ per 4 volte in quanto abbiamo inserito come base 4

➤ aggiungendo alla terza la quarta riga di asterischi ottenendo

```
b=int(input('Base: '))
h=int(input('Altezza: '))
for i in range(h): # ciclo per le righe
    for j in range(b): # ciclo per le colonne
        print('* ', end=' ')
    print()
```

```
* * * *
* * * *
* * * *
* * * *
```

Soluzione n°6.1

➤ Passo 5

➤ i=4 per le righe

➤ entra nel secondo con j=0 e ripete l'istruzione

```
print('* ', end=' ')
```

➤ per 4 volte in quanto abbiamo inserito come base 4

➤ aggiungendo alla quarta la quinta riga di asterischi ottenendo

```
b=int(input('Base: '))
h=int(input('Altezza: '))
for i in range(h): # ciclo per le righe
    for j in range(b): # ciclo per le colonne
        print('* ', end=' ')
    print()
```

```
* * * *
* * * *
* * * *
* * * *
* * * *
```

Le matrici



Differenza tra una variabile e un vettore e una matrice

Variabile
Int **a**

2

Vettore «**vet**»

Posizione 0	2	vet[0]
Posizione 1	4	vet[1]
Posizione 2	6	vet[2]
Posizione 3	9	vet[3]

Matrice «**mat**»

2	2	2	mat[0][0]
4	4	4	mat[1][1]
6	6	6	mat[2][0]
9	9	9	mat[3][2]

- Una matrice è una lista/vettore di lista/vettore



colonne

3	7	10	0
1	3	11	2
5	8	9	24

righe

Dichiarazione di una matrice in Python

- Matrice inserita nel codice

```
mat=[[1,2],[3,4],[5,6]] #matrice 3x2
h=3 #numero di righe
b=2 #numero di colonne
for i in range(h):           # ciclo per le righe
    for j in range(b):       # ciclo per le colonne
        print(mat[i][j], end=' ')
print()
```

Inserimento
dei valori
nella matrice
nel codice

visualizza i
valori nella
matrice

Dichiarazione di una matrice in Python

- Matrice inserita dall'utente

```
h=2 #numero di righe
b=2 #numero di colonne
mat=[]
for i in range(h):      # ciclo per le righe
    n=[]
    for j in range(b):  # ciclo per le colonne
        x=int(input("inserimento elemento nella riga n° "+str(i)+" e colonna n° "+str(j)))
        n.append(x)
        mat.append(n)

print(mat, end=' ')
print()
```

Inserimento i
valori nella
matrice

- I dati sono inseriti dall'utente

```
inserimento elemento nella riga n° 0 e colonna n° 01
inserimento elemento nella riga n° 0 e colonna n° 12
inserimento elemento nella riga n° 1 e colonna n° 03
inserimento elemento nella riga n° 1 e colonna n° 14
[[1, 2], [3, 4]]
```

Esercitazioni n°7 (matrici)

1. Realizziamo un algoritmo per il calcolo del valore massimo di una matrice 3x2, dati i seguenti valori, `mat=[[1,2],[3,4],[5,6]]`
2. Popolare una matrice 4x3. Dopo l'inserimento visualizzare tutti gli elementi della lista con il relativo indice. Dopo trovare il valore massimo tra quelli inseriti nella lista.
3. L'utente dopo aver inserito una matrice 3x2. Visualizza i valori della matrice e la somma, il prodotto e la media.
4. L'utente dopo aver inserito una matrice 3x3, visualizzare il valore minimo.
5. L'utente dopo aver inserito i valori di una matrice 4x4. Visualizzare solo i numeri pari e la loro somma e il loro prodotto.
6. Caricare una matrice 3x3 dei valori casuali utilizzando il metodo random, visualizzare tutti gli elementi con la relativa posizione, e i valori multipli di 3 con la relativa posizione.

Soluzione n°7.1

```
mat=[[1,2],[3,4],[5,6]] #matrice 3x2
h=3 #numero di righe
b=2 #numero di colonne
max=0
for i in range(h):      # ciclo per le righe
    for j in range(b):  # ciclo per le colonne
        if (mat[i][j]>max):
            max=mat[i][j]
print("il valore max è",max)
print()
```

Soluzione n°7.2

```
h=2 #numero di righe
b=2 #numero di colonne
mat=[]
for i in range(h):      # ciclo per le righe
    n=[]
    for j in range(b):  # ciclo per le colonne
        x=int(input("inserimento elemento nella riga n° "+str(i)+" e colonna n° "+str(j)))
        n.append(x)
    mat.append(n)
print(mat, end=' ')
print()
max=0
for i in range(h):      # ciclo per le righe
    for j in range(b):  # ciclo per le colonne
        if (mat[i][j]>max):
            max=mat[i][j]
print("il valore max è",max)
print()
```

