# Algoritmi

# Risoluzione di un problema

Descrizione di un **PROBLEMA** → Individuazione di un **ALGORITMO**

# Algoritmo

Sequenza <u>FINITA</u> di mosse che risolve <u>IN UN TEMPO FINITO</u> una ***classe di*** **PROBLEMI** ovvero

- Deve essere applicabile a un qualsiasi insieme di dati del **dominio**
- Deve produrre risultati appartenenti al **codominio**

# Caratteristiche di un algoritmo

**ESEGUIBILITÀ**: ogni azione deve essere *eseguibile* da parte dell'esecutore dell'algoritmo in un tempo finito

**NON-AMBIGUITÀ**: ogni azione deve essere *univocamente interpretabile* dall'esecutore

**TERMINAZIONE**: il numero totale di azioni da eseguire, per ogni insieme di dati di ingresso, deve essere finito

**DETERMINISMO**: il flusso di esecuzione per lo stesso insieme di dati in input deve sempre essere lo stesso

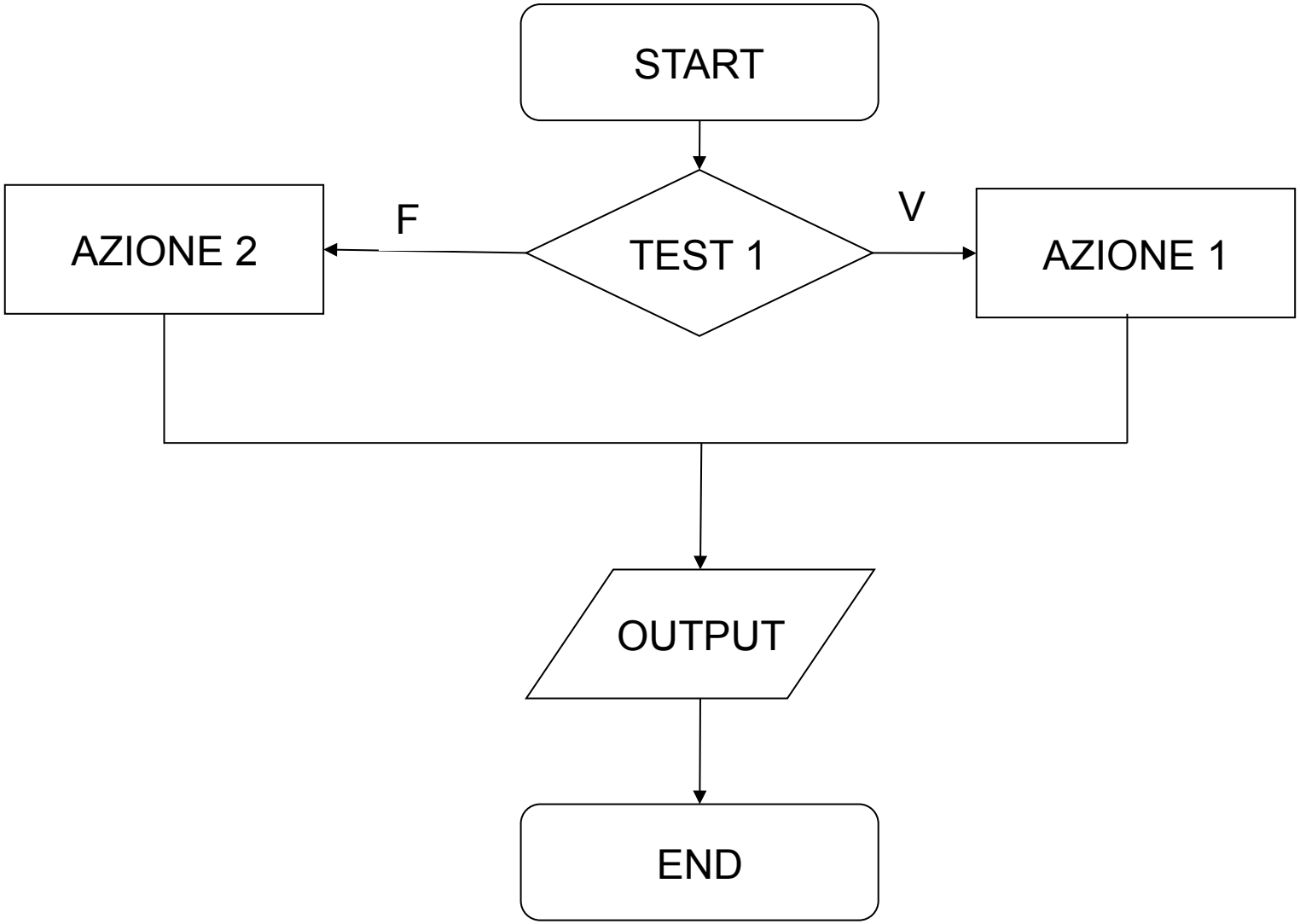
# Rappresentazione di un algoritmo

- Per rappresentare un ALGORITMO occorre rappresentare
  - Le istruzioni necessarie
  - La loro corretta sequenza
- Strumenti
  - Linguaggio naturale
  - Linguaggi (semi)formali
  - ☑Diagrammi di flusso (flow chart)

# Cosa è un DIAGRAMMA DI FLUSSO

- È un **formalismo** che permette di rappresentare graficamente un algoritmo.
- Descrive le azioni da eseguire e il loro ordine di esecuzione.
- Ogni diagramma di flusso è un **grafo**
  - Ogni **nodo** è un'azione elementare e corrisponde ad un simbolo grafico (blocco)
  - Gli **archi** ne determinano la sequenza

# Esempio



# Blocchi fondamentali

START

Blocco **START**

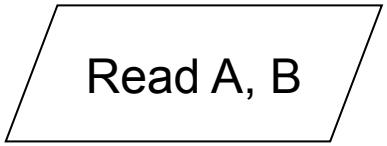
- È il blocco da cui deve iniziare l'algoritmo
- In un diagramma è presente **un solo blocco START**

END

Blocco **END**

- Termina l'esecuzione dell'algoritmo
- In un diagramma è presente **un solo blocco END**

# Blocchi fondamentali




Blocco **READ**

- Riceve dall'unità in ingresso (per esempio la tastiera) tanti valori quante sono le **VARIABILI** specificate all'interno del blocco
- Assegna alle variabili i valori inseriti

Blocco **WRITE**

- Trasmette in uscita (per esempio su video) i valori delle varibili

# Blocchi fondamentali

Azione

Blocco **AZIONE** (o PROCESSO o MOSSA)

- corrisponde ad un'azione elementare:
  - Assegnamento (=)
  - Operazioni aritmetiche (+, -, *, /,…)
  - …

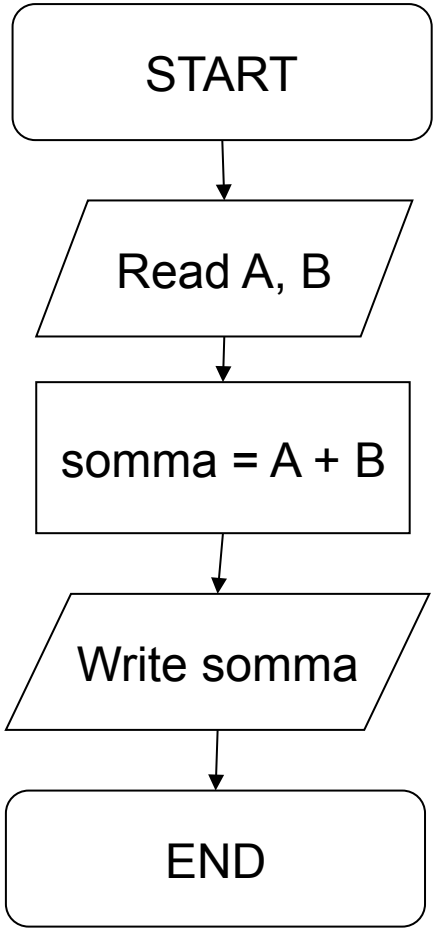
Esempi

A = A + 1

BIG = A

# Esercizio

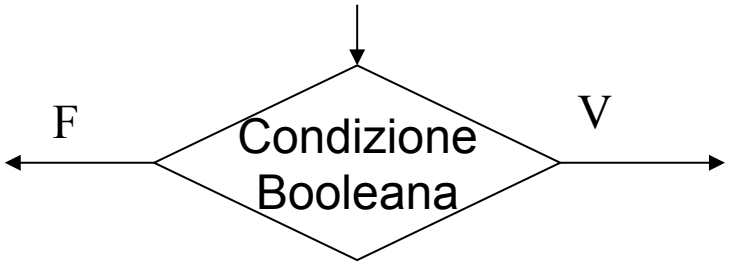
Stampare la somma di due numeri reali dati



# Altri esercizi

- Stampare l'area/il perimetro di un cerchio di raggio dato in ingresso da tastiera
- Stampare l'area di un triangolo isoscele data la lunghezza della base e due lati uguali

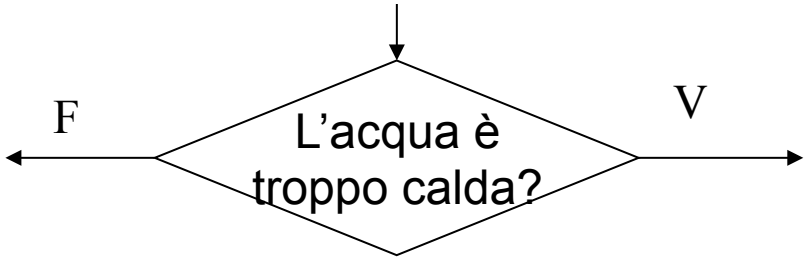
# Blocchi fondamentali

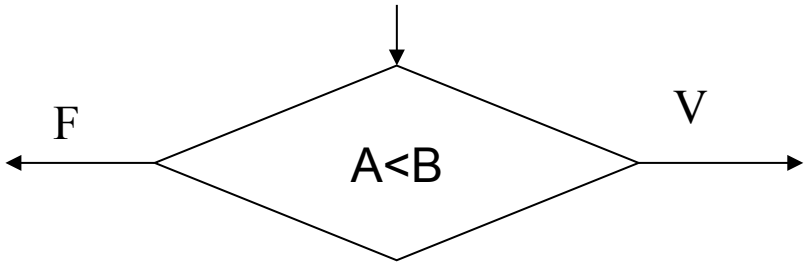



Blocco **TEST**

- Al blocco è associato un test il cui risultato è booleano (VERO/FALSO);
- Se la condizione è vera si procede sul ramo etichettato con V
- Se la condizione è falsa si procede sul ramo etichettato con F

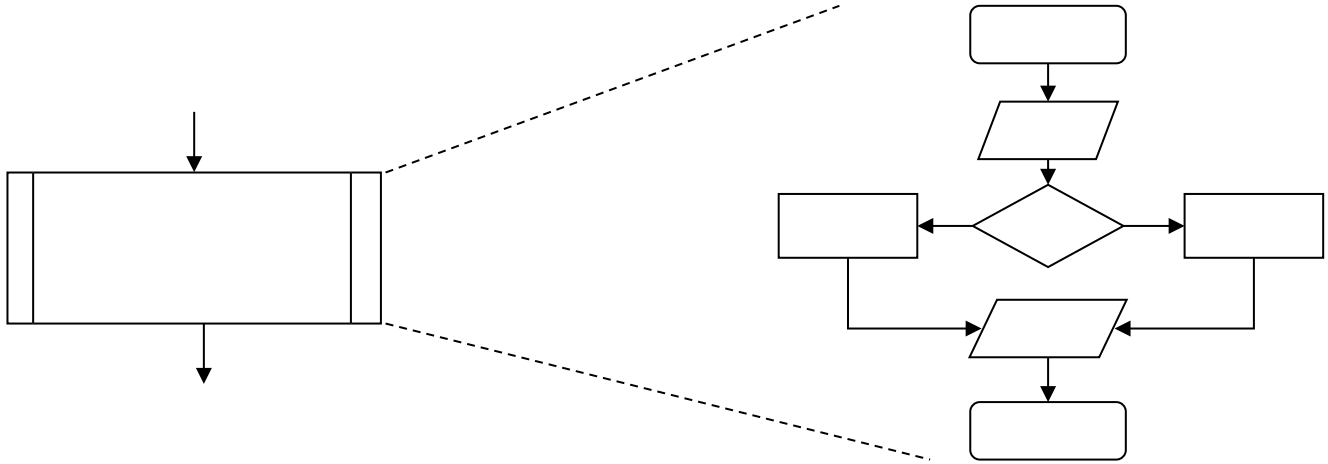
Esempi



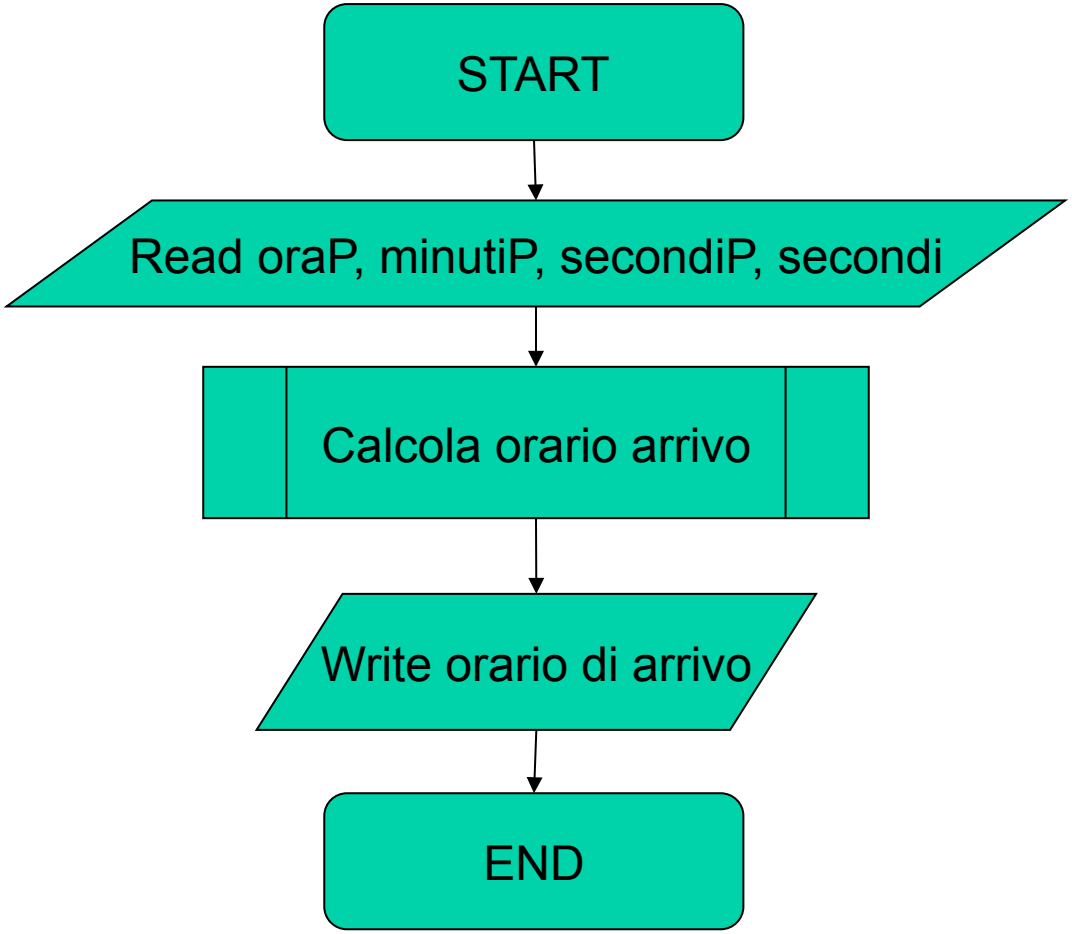
# Blocchi fondamentali

- Blocco **PROCESSO**
  - Indica una sequenza di azioni per ottenere un particolare risultato.
  - È utile per rendere più compatto il diagramma di flusso.



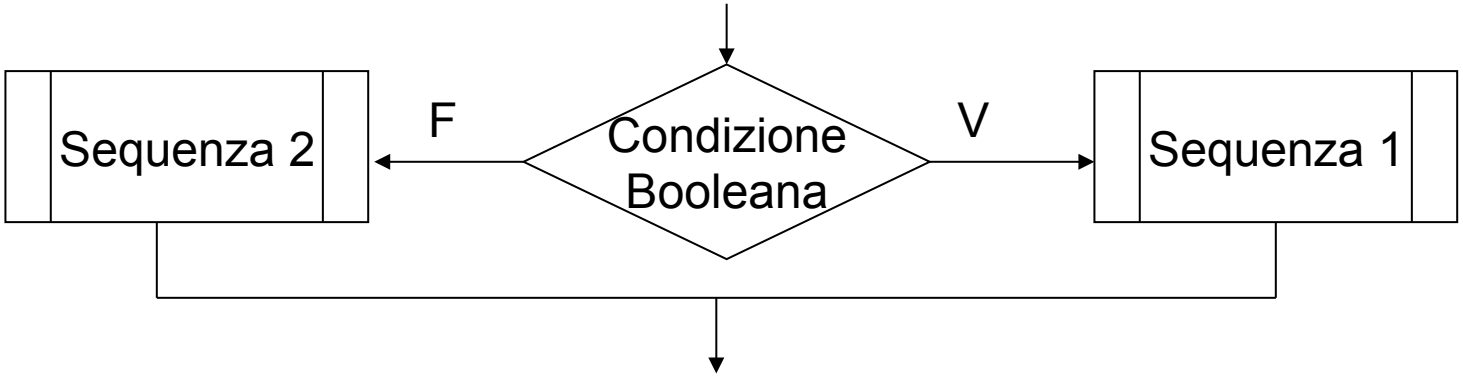
# Esercizio

- Dato un orario di partenza e un tempo di percorrenza in secondi, calcolare l'orario di arrivo

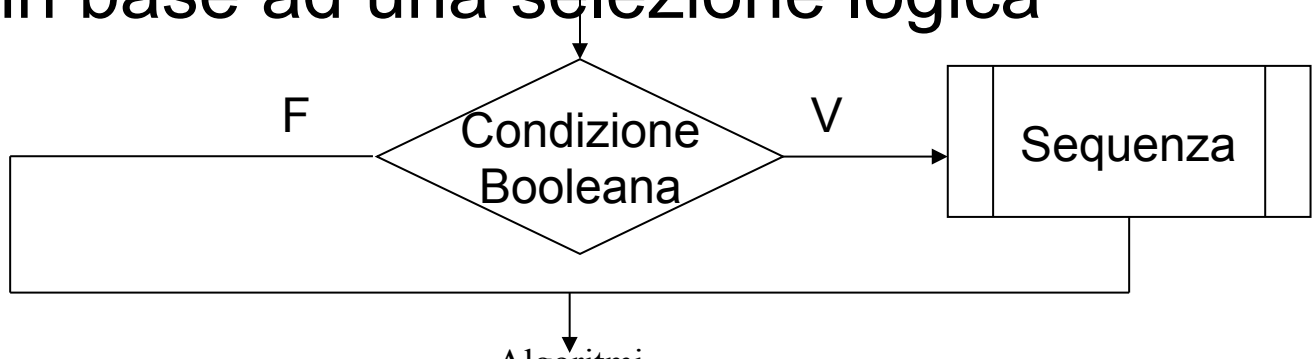


# Strutture condizionali IF-THEN(-ELSE)

- Permette di scegliere tra 2 sequenze di azioni in base ad una selezione logica

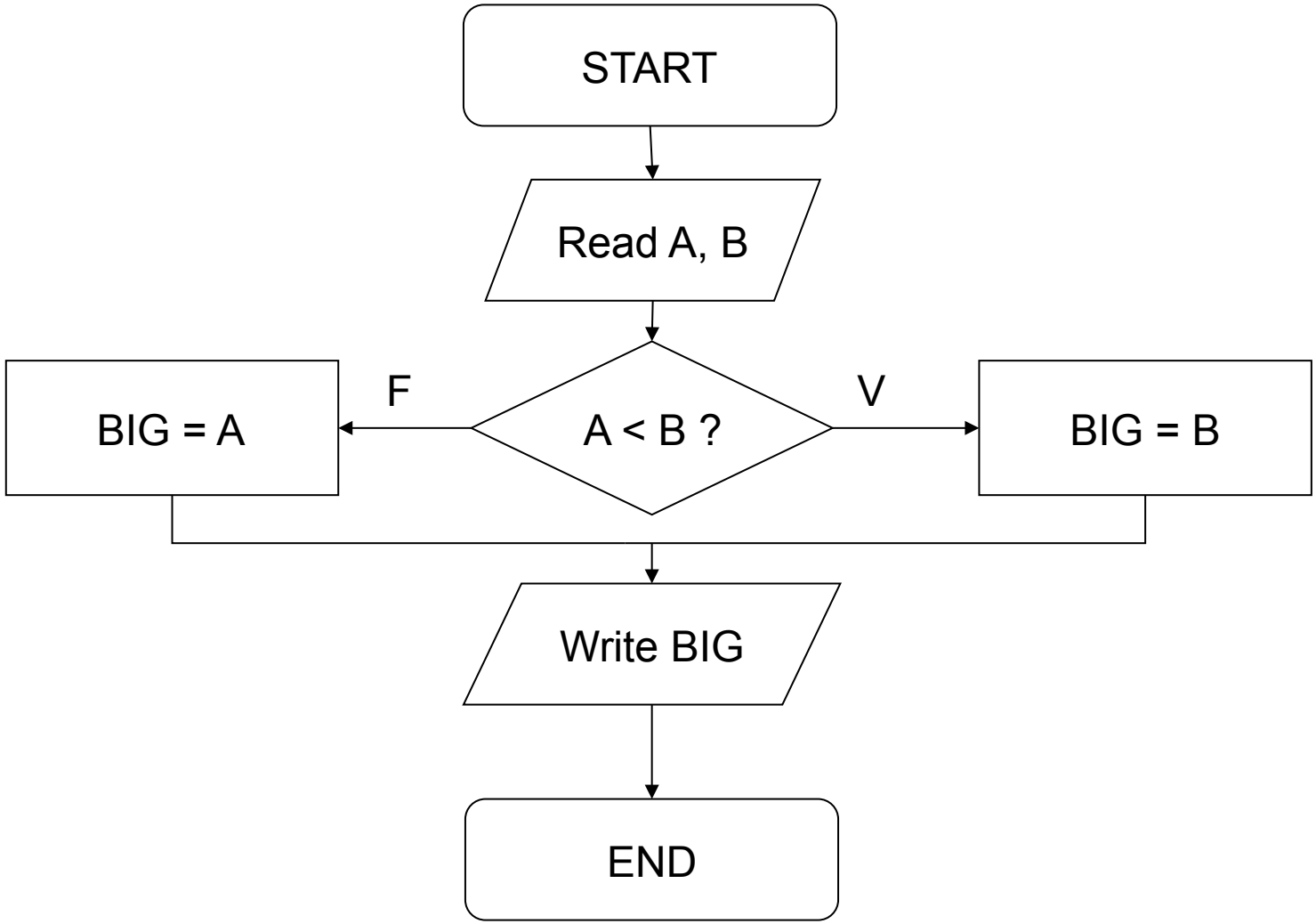



- Permette scegliere se eseguire una sequenza di azioni in base ad una selezione logica



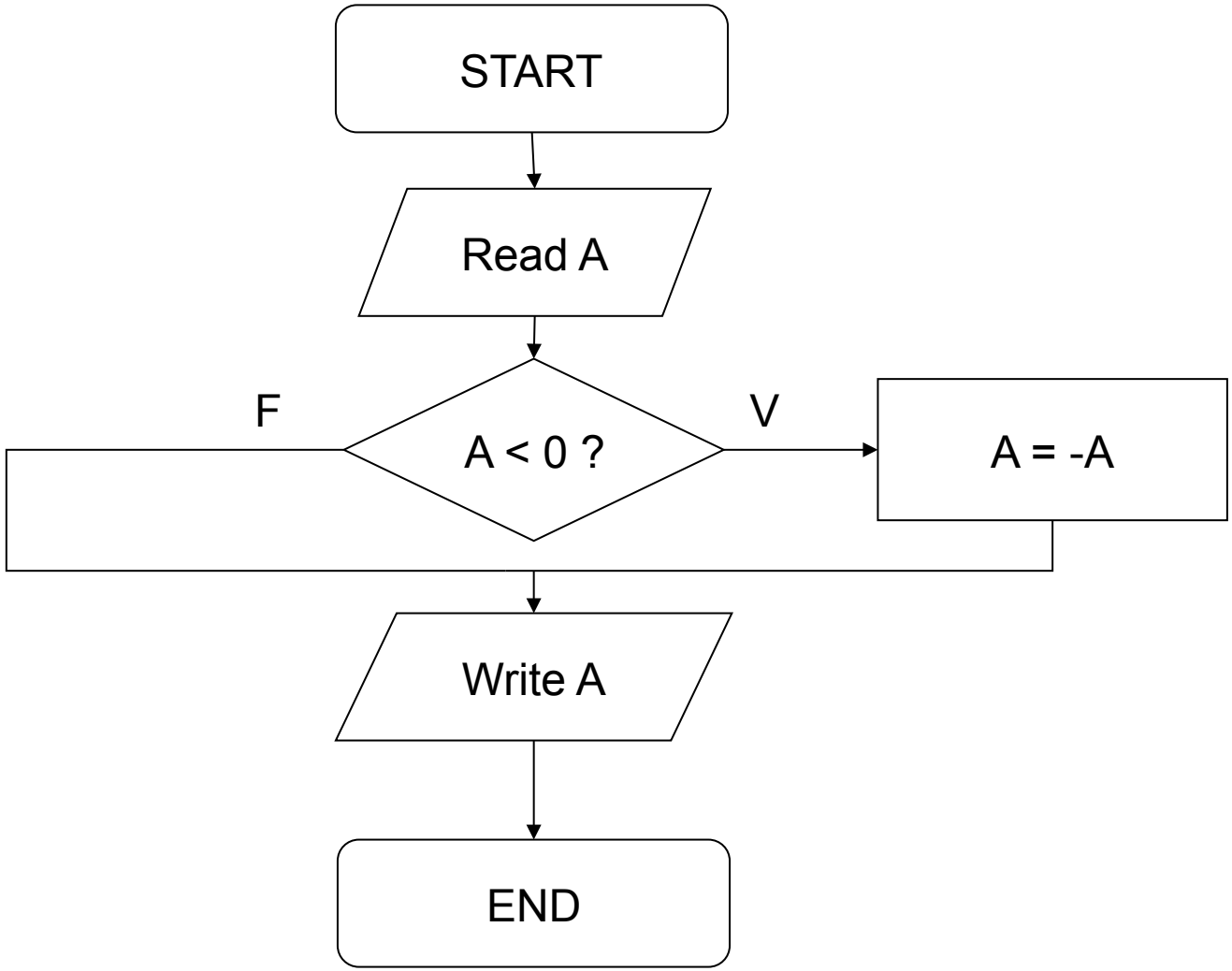
# Esercizio

Trovare il massimo di due valori dati



# Esercizio

Calcolare il valore assoluto di un numero dato

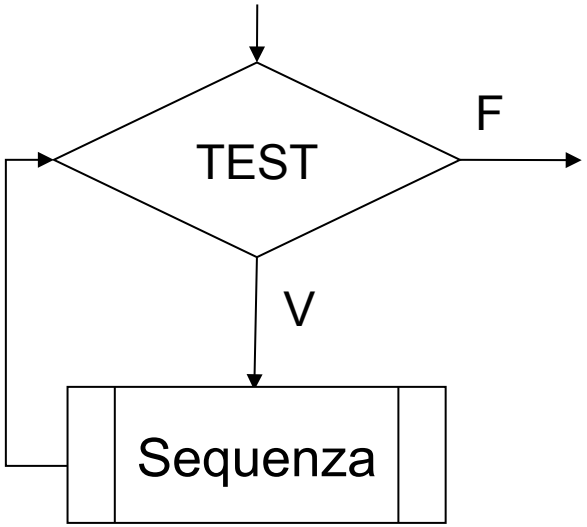


# Esercizi

- Calcolare il minimo di due numeri dati
- Verificare se un numero dato è pari
- Calcolare il massimo di tre numeri dati

# Struttura iterativa WHILE

- Esegue la stessa sequenza di azioni fintanto che la condizione risulta vera.



# Esercizio

Calcolare la somma di N numeri forniti in input

**OSSERVAZIONI**

E' necessario calcolare la somma in modo incrementale perchè i valori vengono inseriti uno alla volta

**DEFINIZIONE PER INDUZIONE**

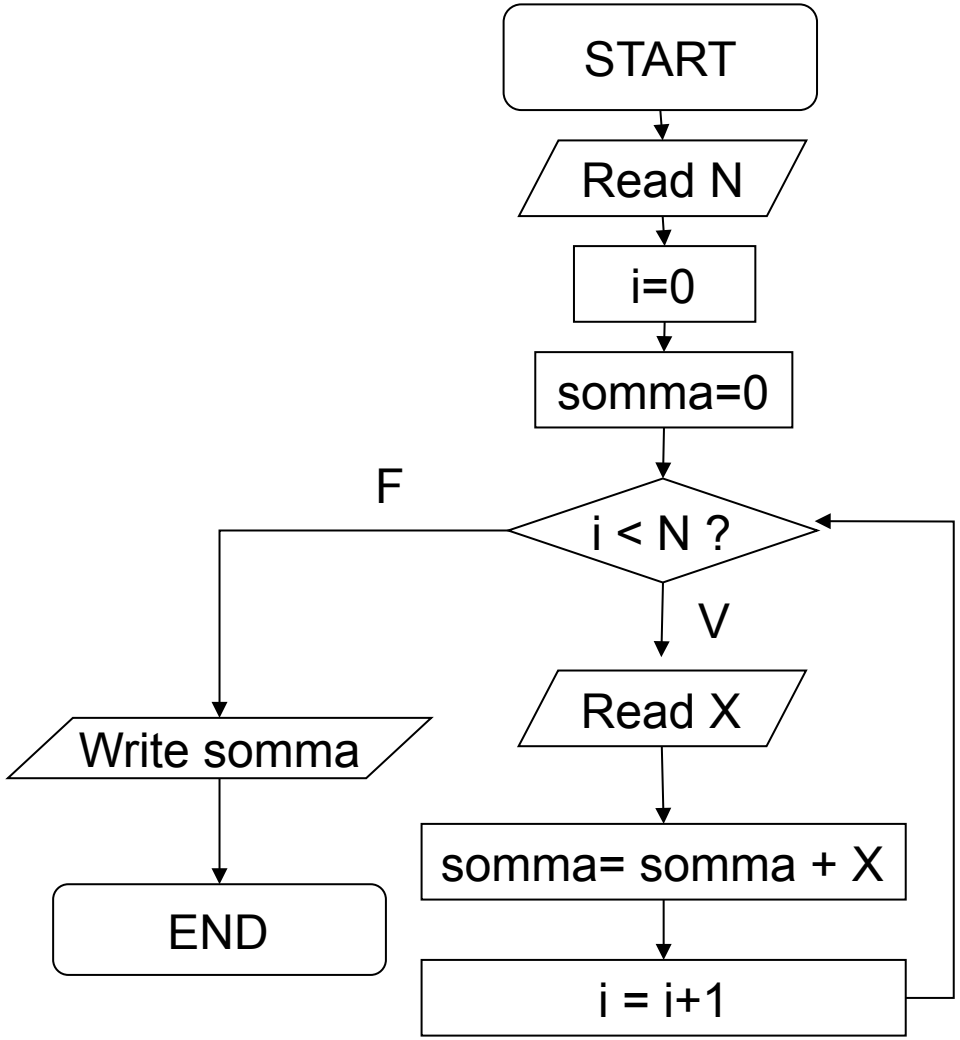
CASO BASE: La somma di 0 numeri è 0

somma({})=0

CASO INDUTTIVO: La somma di N+1 numeri $\{X_1,..,X_{n+1}\}$ è la somma della somma di $\{X_1,..,X_n\}$ e $X_{n+1}$

$\text{somma}(\{X_1,..,X_{n+1}\})=\text{somma}(\{X_1,..,X_n\})+X_{n+1}$

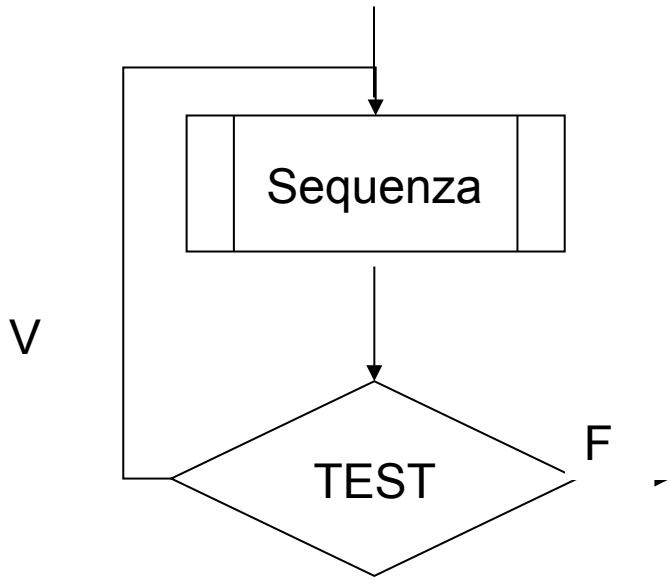
# Esercizio (cont.)



# Struttura iterativa DO…WHILE

- Esegue la sequenza di azioni fino a quando la condizione risulta vera.

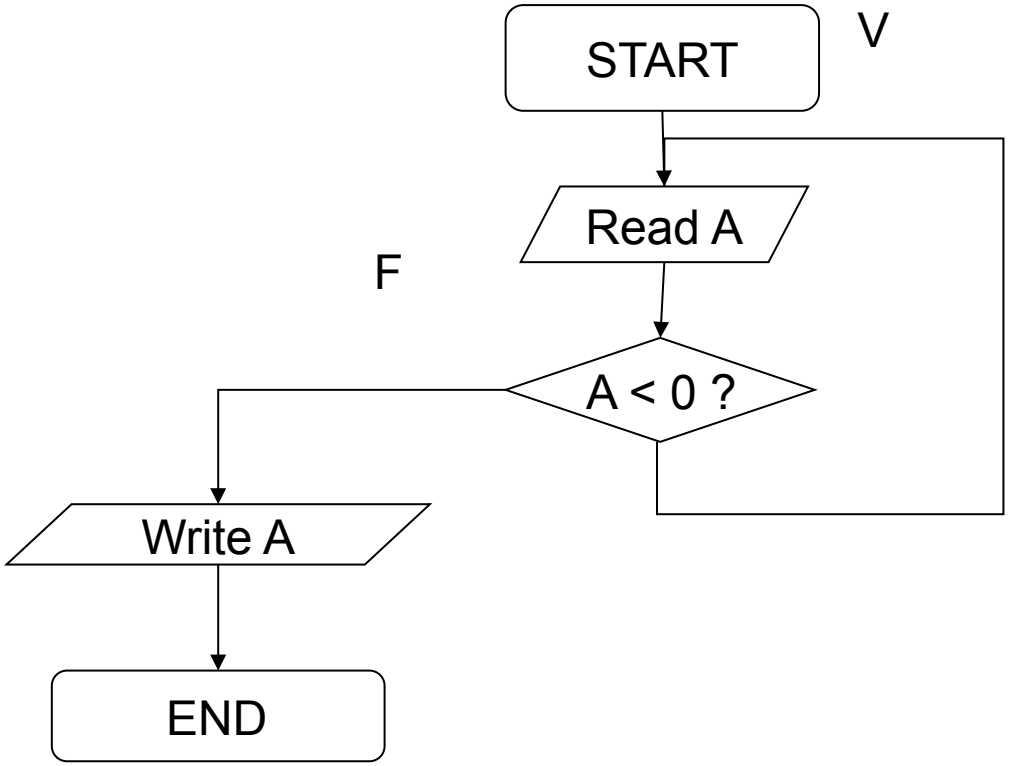
Nota: l'azione è eseguita almeno una volta.



# Esercizio

Leggere un valore numerico POSITIVO da tastiera.

N.B. E' necessario chiedere il valore all'utente fintantoché questo non è positivo



# Esercizi

- Rappresentare tramite i flow-chart l'algoritmo di Euclide

# Esercizio

- Scrivere un algoritmo che calcoli le radici di un'equazione di secondo grado $Ax^2+Bx+C=0$.

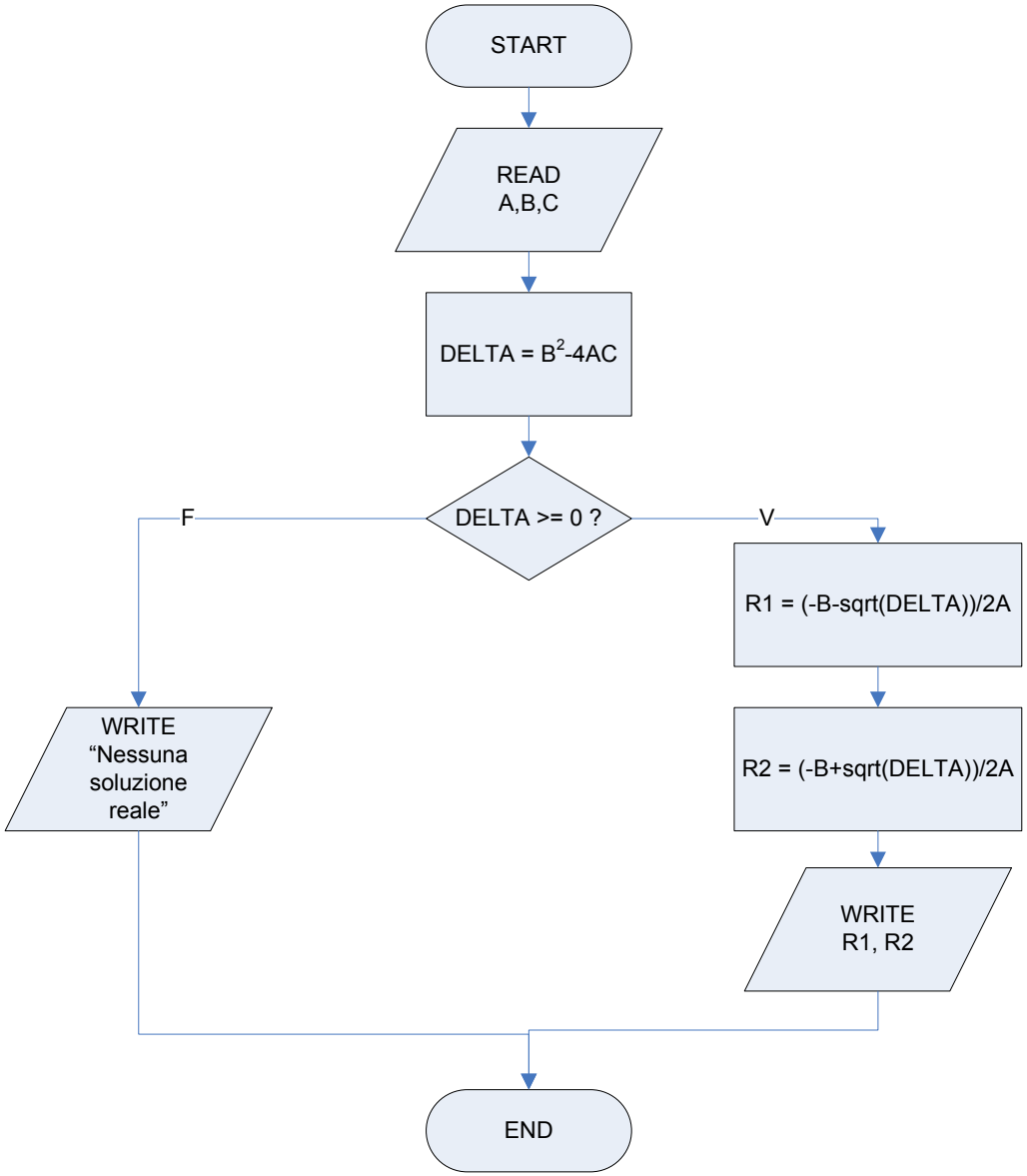
N.B. si supponga di disporre della funzione sqrt(.) che dato un valore in ingresso ne calcola la radice quadrata

**OSSERVAZIONI**

Dato il valore $\Delta = B^2 - 4AC$ si ricorda che se Δ è negativo allora non ci sono soluzioni reali altrimenti ci sono 2 soluzioni reali

$$soluzioni = \frac{-B \pm \sqrt{\Delta}}{2A}$$

# Esercizio


START
READ
A,B,C
DELTA = B2-4AC
DELTA >= 0 ?
F
V
R1 = (-B-sqrt(DELTA))/2A
R2 = (-B+sqrt(DELTA))/2A
WRITE
R1, R2
WRITE
“Nessuna
soluzione
reale”
END

# Esercizio

- Scrivere un algoritmo che ricerchi il minimo di K numeri forniti in input. K è fornito in input dall'utente ed è maggiore o uguale a 1.

**OSSERVAZIONI**

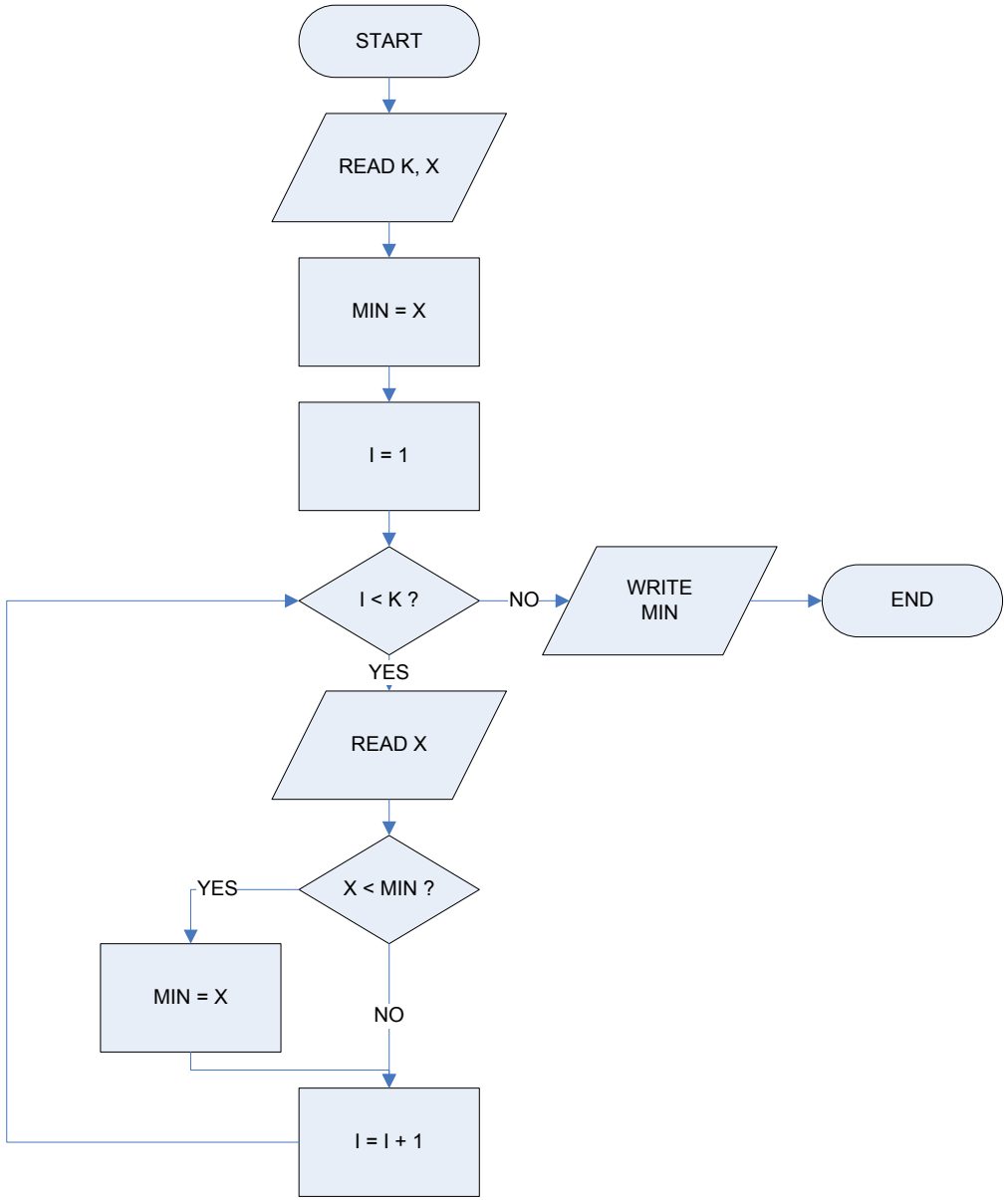
E' necessario definire il minimo per induzione perchè i valori vengono inseriti uno alla volta

CASO BASE: minimo({X})=X

CASO INDUTTIVO:

minimo($\{X_1,..,X_{n+1}\}$)= minore tra minimo($\{X_1,..,X_n\}$) e $X_{n+1}$

# Esercizio


START
READ K, X
MIN = X
I = 1
I < K ?
NO
WRITE
MIN
END
YES
READ X
X < MIN ?
YES
MIN = X
NO
I = I + 1

# Esercizio

- Scrivere un algoritmo che esegua la moltiplicazione di due numeri interi mediante somma.

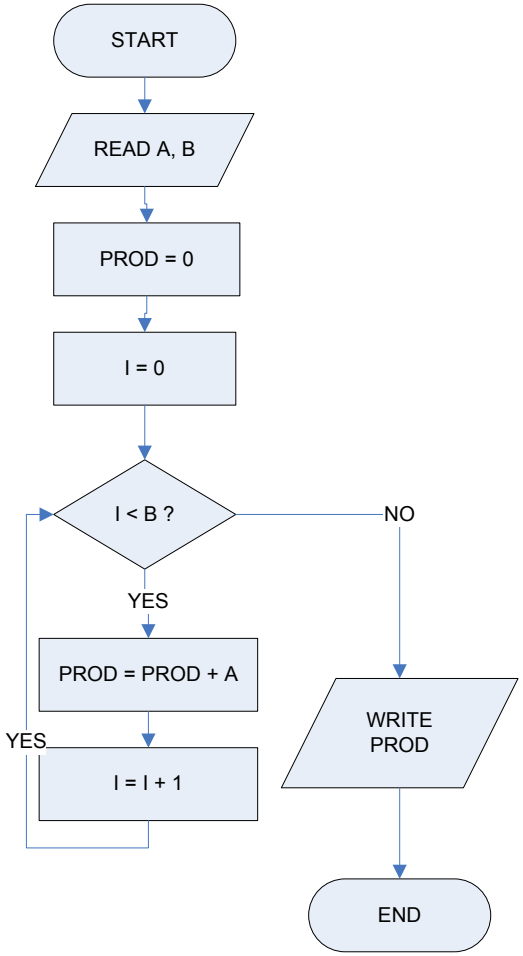
**OSSERVAZIONI**

Definizione per induzione

CASO BASE: A*0=0

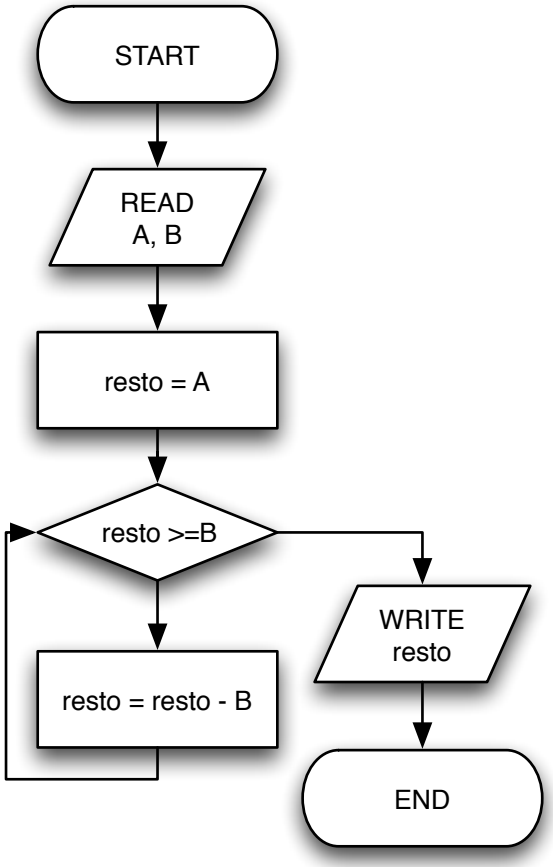
CASO INDUTTIVO: A*(B+1) = (A*B)+A

# Esercizio


START
READ A, B
PROD = 0
I = 0
I < B ?
NO
YES
PROD = PROD + A
YES
I = I + 1
WRITE
PROD
END

# Esercizio

- Scrivere un algoritmo per il calcolo del resto della divisione tra due valori interi



# Altri esercizi

- Scrivere un algoritmo che calcoli la media di N valori interi dati in input
- Modificare l'esercizio precedente assumendo che i numeri in ingresso siano dei voti (ovvero dei numeri compresi tra 18 e 30)
- Scrivere un algoritmo che calcoli il quoziente e il resto di una divisione tra interi
- Scrivere un algoritmo che calcoli l'intersezione di due intervalli [I1,S1] e [I2,S2], dati gli estremi degli intervalli stessi
- Scrivere un algoritmo che calcoli l'unione di due intervalli intersecanti [I1,S1] e [I2,S2], dati gli estremi degli intervalli stessi
- Scrivere un algoritmo che calcoli i divisori di un numero dato
- Scrivere un algoritmo che verifichi se il numero dato in ingresso è primo
- Scrivere un algoritmo che calcoli il fattoriale di un numero dato

# Algoritmi equivalenti

**Dato un PROBLEMA, nella maggior parte dei casi è possibile individuare PIÙ ALGORITMI che lo risolvono!**

Due algoritmi si dicono **EQUIVALENTI** quando:

- hanno lo stesso dominio di ingresso
- hanno lo stesso dominio di uscita
- in corrispondenza degli stessi valori nel dominio di ingresso *producono gli stessi valori* nel dominio di uscita

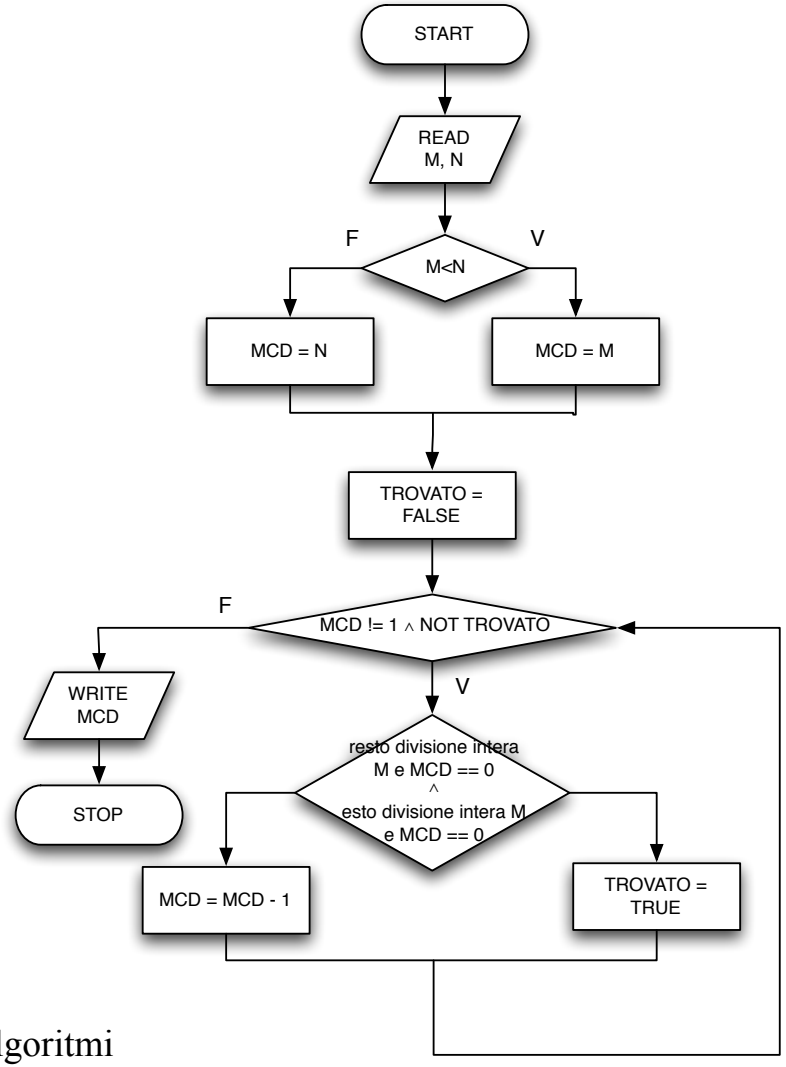
Due algoritmi equivalenti:

- forniscono lo **STESSO RISULTATO**
- ma possono avere **DIVERSA EFFICIENZA**
- e possono essere **PROFONDAMENTE DIVERSI!**

# Esempio

Calcolo del Massimo Comun Divisore (MCD) fra due interi M ed N

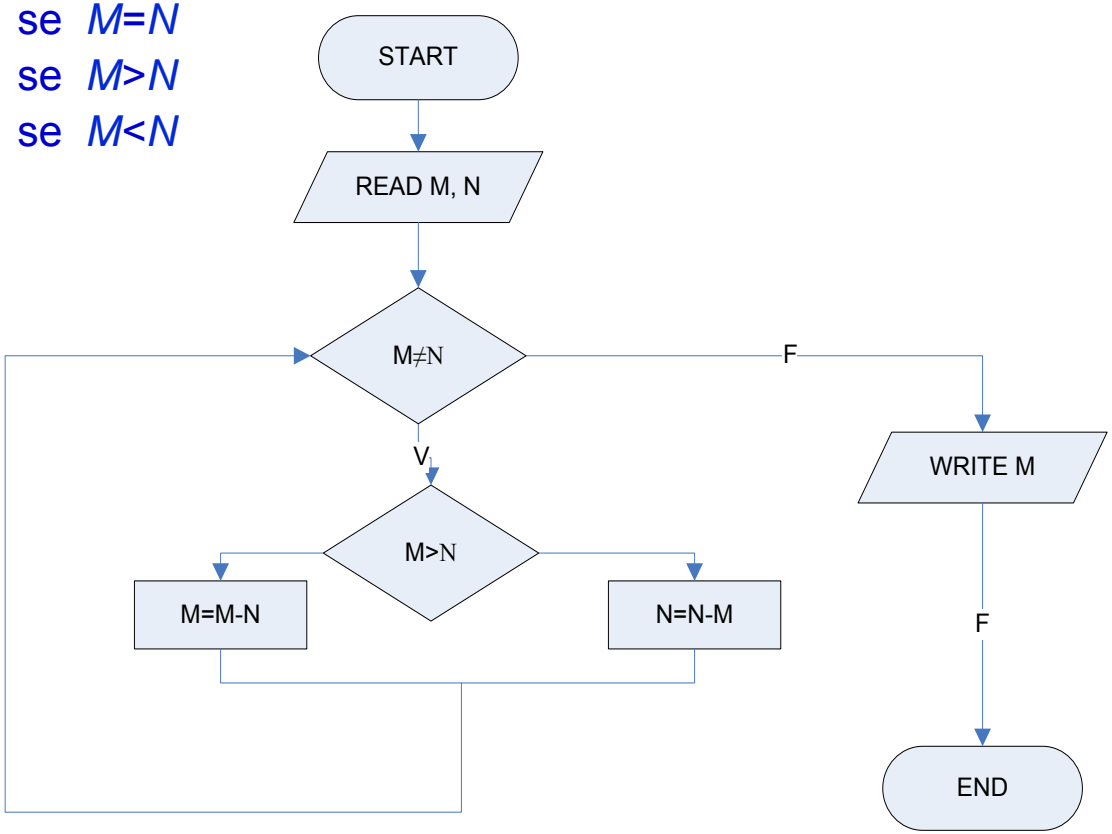
## Algoritmo n° 1



# Esempio

## Algoritmo n° 2 (metodo di Euclide)

$$\text{MCD}(M,N) = \begin{cases} M \text{ (oppure N)} & \text{se } M=N \\ \text{MCD}(M-N,\ N) & \text{se } M>N \\ \text{MCD}(M,\ N-M) & \text{se } M<N \end{cases}$$



# Esempio (*cont.*)

**Calcolare il MCD dei numeri 12 e 21**

<u>ALGORITMO 1</u>: $M \leftarrow 12$, $N \leftarrow 21$

$MCD \leftarrow 12$

$MCD \leftarrow 11$

…

$MCD \leftarrow 3$ **MCD = 3**

<u>ALGORITMO 2:</u> $M \leftarrow 12$, $N \leftarrow 21$

$N \leftarrow (N-M) = 9$

$M \leftarrow (M-N) = 3$

$N \leftarrow (N-M) = 6$

$N \leftarrow (N-M) = 3$ **MCD = 3**

# MANCA LA CODIFICA!

## Perché?

- Bisogna conoscere la **SINTASSI** di un linguaggio di programmazione
- Bisogna conoscere la **SEMANTICA** di un linguaggio di programmazione
- Bisogna conoscere un **AMBIENTE DI PROGRAMMAZIONE**

Quale linguaggio? Quale ambiente di programmazione?