La programmazione

CORSO DI INFORMATICA

Maria Grazia Celentano

INFORMATICA E PROGRAMMAZIONE

- L'Informatica è definita come la **Scienza della Rappresentazione** e dell'**Elaborazione dell'informazione** o, in altri termini, lo **studio delle strutture dati che rappresentano le informazioni e degli algoritmi che le elaborano**.
- E' necessario distinguere tra **DATO** e **INFORMAZIONE legata al dato.**

DATI e INFORMAZIONI

- Una sequenza di dati non è necessariamente fonte d'informazione.
- Supponiamo di avere il seguente elenco di numeri: 18, 20, 21, 24, 23, 25, 28, 18, 20, 21, 24, 25, 27, 28, 18, 21, 21, 24, 24, 25, 29, 18, 20, non è immediatamente associato ad alcuna informazione.
- L'elenco potrebbe rappresentare le temperature registrate nel mese di luglio così come i voti d'esame di altrettanti studenti.
- **I dati danno luogo a informazione nel momento in cui sono collocati in un contesto ben preciso**.
- In altri termini ***se al dato è possibile associare un significato ben preciso esso diventa informazione, altrimenti rimane un mero dato*.**

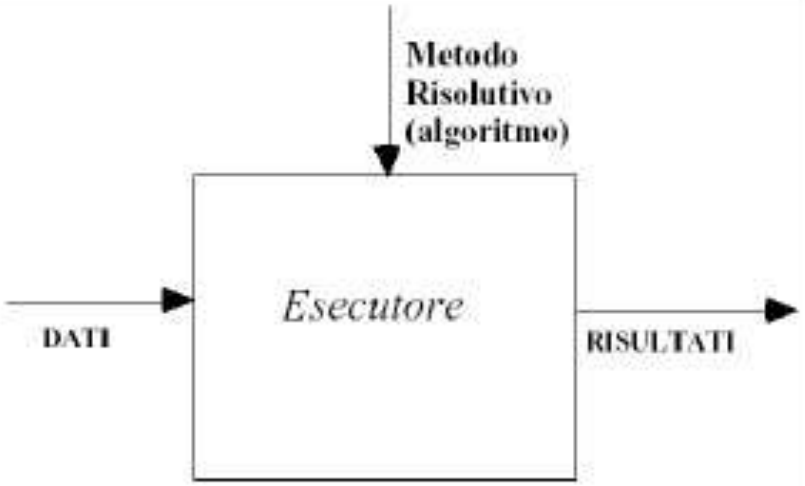
RAPPRESENTAZIONE DEI DATI

- L'Informatica si occupa della rappresentazione e dell'elaborazione dei dati (o delle informazioni). **I dati devono ovviamente essere rappresentati in maniera conveniente**. Una rappresentazione conveniente della precedente sequenza di numeri potrebbe ad es. essere quella basata su numeri naturali. Tale rappresentazione non sarebbe conveniente invece se i numeri rappresentassero anche valori negativi (per esempio se le temperature fossero relative al mese di gennaio).
- Una volta convenientemente rappresentati, i dati possono essere elaborati (es. calcolo della temperatura media).
- **I dati vengono *processati* attraverso un algoritmo al fine di ottenere un risultato**: si sommano i valori relativi alle temperature e si divide per il loro numero.

ALGORTIMO

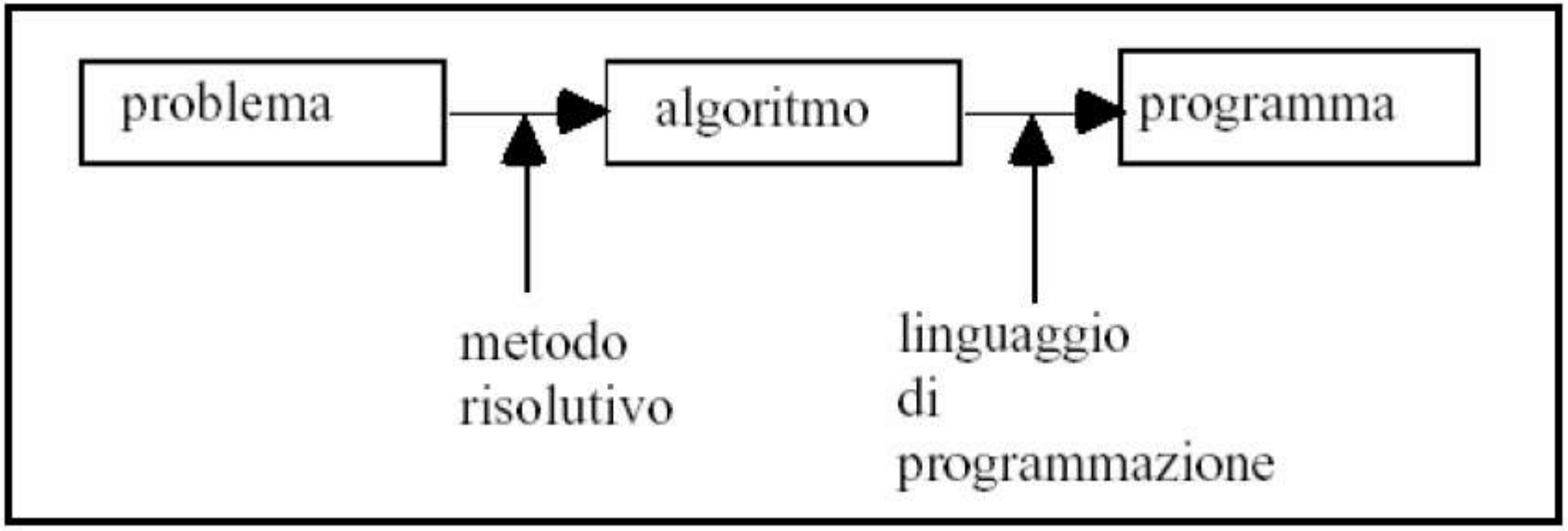
- Un *algoritmo* può essere informalmente definito come una ***sequenza finita di "mosse" che risolve in un tempo finito un determinato problema***. L'esecuzione delle azioni nell'ordine specificato dall'algoritmo consente di ottenere, a partire dai dati di ingresso, i risultati che rappresentano la risoluzione del problema.

Esecutore: una *macchina astratta* capace di *eseguire le azioni* specificate dall'algoritmo.



PROGRAMMAZIONE

- Un algoritmo deve essere definito in modo che possa essere **interpretato ed eseguito correttamente dalla macchina**. A tal proposito esistono differenti ***linguaggi di programmazione*** che consentono di definire le istruzioni dell'algoritmo per la loro esecuzione su macchine calcolatrici.



Sviluppo del software

- problema
- idea (soluzione informale)
- algoritmo (soluzione formale)
- programma (traduzione dell'algoritmo in una forma comprensibile da un elaboratore elettronico)
- test (su molti casi, con particolare attenzione ai casi limite)
- documentazione (manuale utente + manuale del programmatore)

Algoritmo

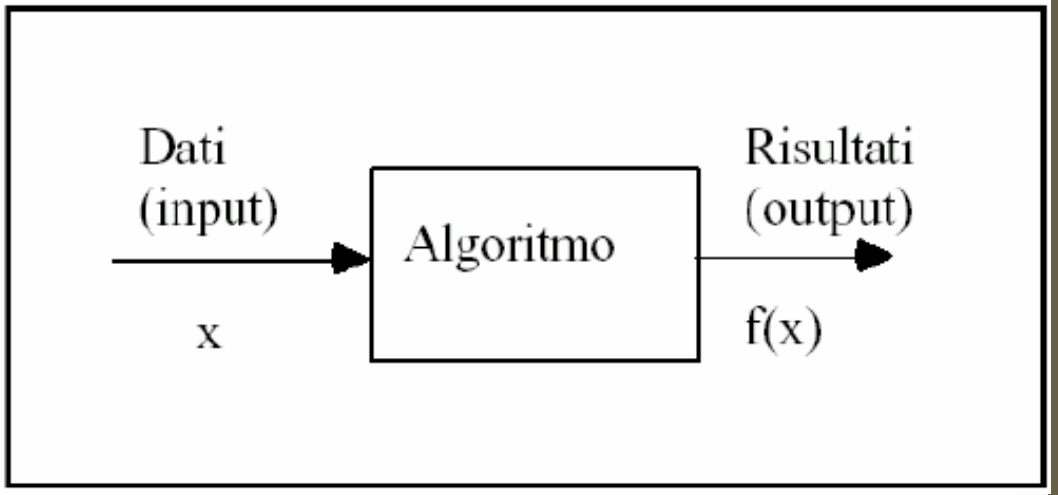
Un algoritmo può essere considerato un insieme di regole per svolgere un dato compito (risolvere un problema).

Il nome deriva dal matematico persiano Muhammad ibn Mūsa 'l-Khwārizmī (780-850).

Un algoritmo deve:

- **terminare in un tempo finito**
- **produrre un effetto osservabile**
- **essere deterministico, ossia produrre gli stessi risultati a partire dalle stesse condizioni iniziali**

ALGORITMI



- Le *proprietà fondamentali* di un algoritmo sono le seguenti:

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

  2. **Non-ambiguità**: ogni azione deve essere u*nivocamente interpretabile* dall'esecutore

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

- Due algoritmi si dicono **equivalenti** quando:

  1. hanno lo stesso dominio di ingresso;

  2. hanno lo stesso dominio di uscita;

  3. in corrispondenza degli stessi valori nel dominio di ingresso *producono gli stessi valori* nel dominio di uscita.

Esecuzione di un algoritmo

Vengono eseguite in sequenza le operazioni che lo costituiscono.

Esistono algoritmi che prevedono:

- **una sequenza di esecuzione unica**
- **sequenze di esecuzione multiple**

Esempio: sequenza di esecuzione unica

Dato il valore di X, calcolare: Y = 5 X + 3

Sequenza di esecuzione:

1. **ricevo il valore di X**
2. **moltiplico X per 5 (sia Z il risultato)**
3. **sommo 3 a Z (sia Y il risultato)**
4. **visualizzo Y**

Esempio: sequenze di esecuzione multiple

Dato il valore di X, calcolare la radice quadrata di X+5.

Sequenza di esecuzione:

1. ricevo il valore di X

2. sommo 5 a X (sia Y il risultato)

3a. se Y è positivo o nullo, calcolo la sua radice quadrata e la visualizzo

3b. se Y è negativo, indico che è impossibile calcolare la sua radice quadrata

Formalizzazione di una soluzione

- **pseudo-linguaggio (o pseudo-codice)**
  - **vantaggi: immediato**
  - **svantaggi: descrizione dell'algoritmo poco astratta, interpretazione più complicata**
- **diagrammi di flusso**
  - **vantaggi: più intuitivi perché usano un formalismo grafico, descrizione dell'algoritmo più astratta**
  - **svantaggi: richiedono l'apprendimento della funzione dei vari tipi di blocco**

Diagrammi di flusso (flow-chart)

- metodo grafico per descrivere in modo formale un algoritmo
- blocchi base per descrivere:
  - azioni
  - decisioni (solo binarie, ossia della logica Booleana)
- archi orientati per descrivere la sequenza di svolgimento delle azioni

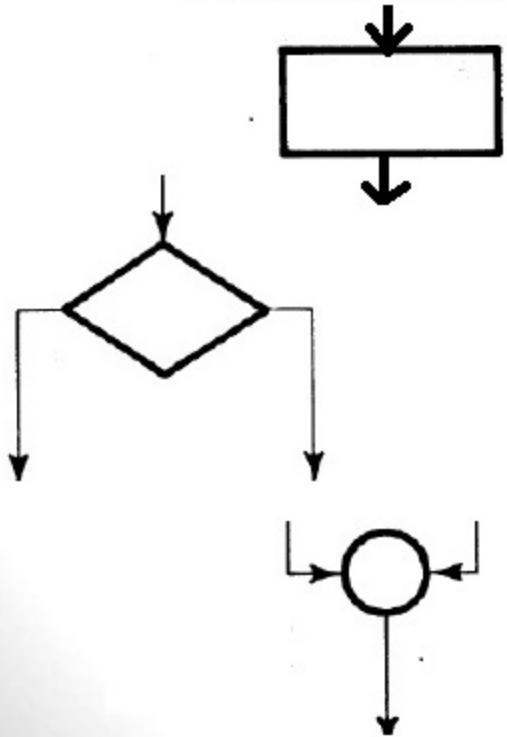
TIPOLOGIE ELEMENTARI DI BLOCCHI

— uno che indica una struttura operativa: *NODO DI PROCESSO*

— uno che indica una struttura di controllo: *NODO DECISIONALE*

— uno che indica una giunzione tra parti diverse di uno stesso algoritmo: *NODO COLLETTORE*

In figura 1.5.1 sono rappresentate queste tre unità elementari.



NODO DI PROCESSO

NODO DECISIONALE

NODO COLLETTORE

TIPOLOGIE ELEMENTARI DI BLOCCHI

Le funzioni associate ai singoli blocchi sono:

— **NODO DI PROCESSO** : *una trasformazione dei dati inseriti;*

— **NODO DECISIONALE** : *un test (o prova) sulla condizione inserita nel blocco che produce una scelta, a seconda che sia verificata (vero) o meno (falso) la condizione. Una decisione può essere attiva o passiva. Nella decisione passiva si testa il risultato di una operazione che viene eseguita precedentemente; in quella attiva vengono eseguite particolari operazioni logiche o aritmetiche per stabilire il test di condizione.*

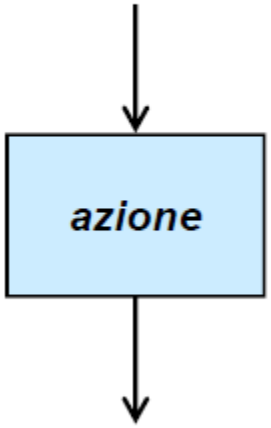
— **NODO COLLETTORE** : *non produce nulla. Esso è solo una giunzione che possiede due entrate ed una sola uscita.*

TIPOLOGIE ELEMENTARI DI BLOCCHI

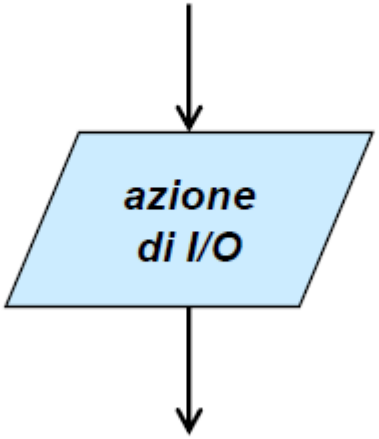
Blocchi di inizio e fine

TIPOLOGIE ELEMENTARI DI BLOCCHI

Blocco di azione



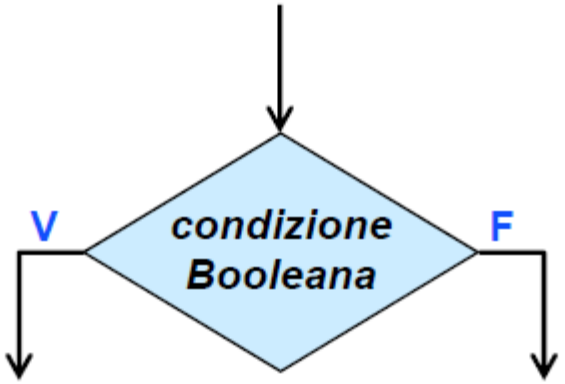
Blocco di Input/Output



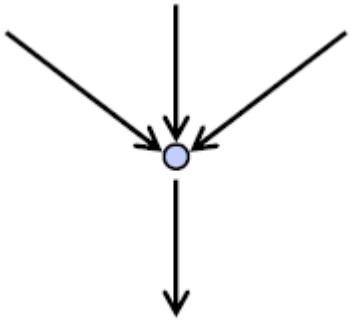
TIPOLOGIE ELEMENTARI DI BLOCCHI

Blocco di inizializzazione

Blocco di decisione binaria



Connettore

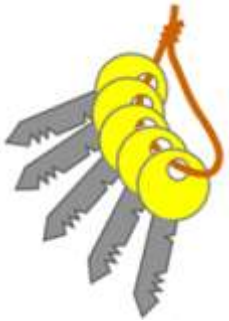


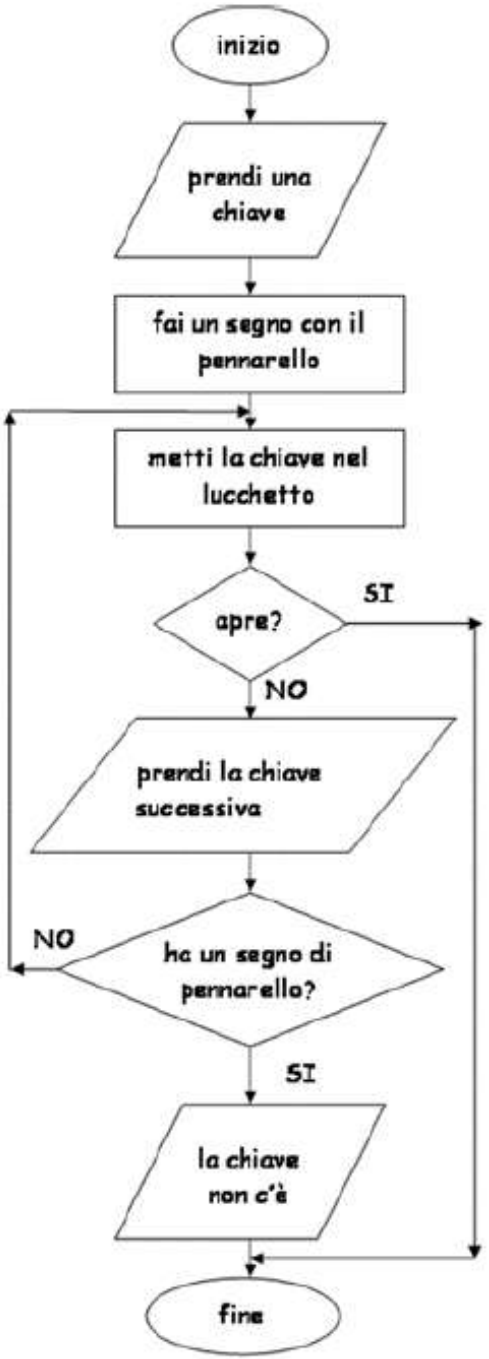
Regole

- **uno ed un solo blocco START**
- **uno ed un solo blocco STOP**
- **tutti gli archi devono avere origine e fine in un blocco**

ESEMPIO

- Un esempio di flow chart relativo al problema di **trovare in un mazzo di chiavi quella che apre un lucchetto** è il seguente:





Diagrammi di flusso strutturati

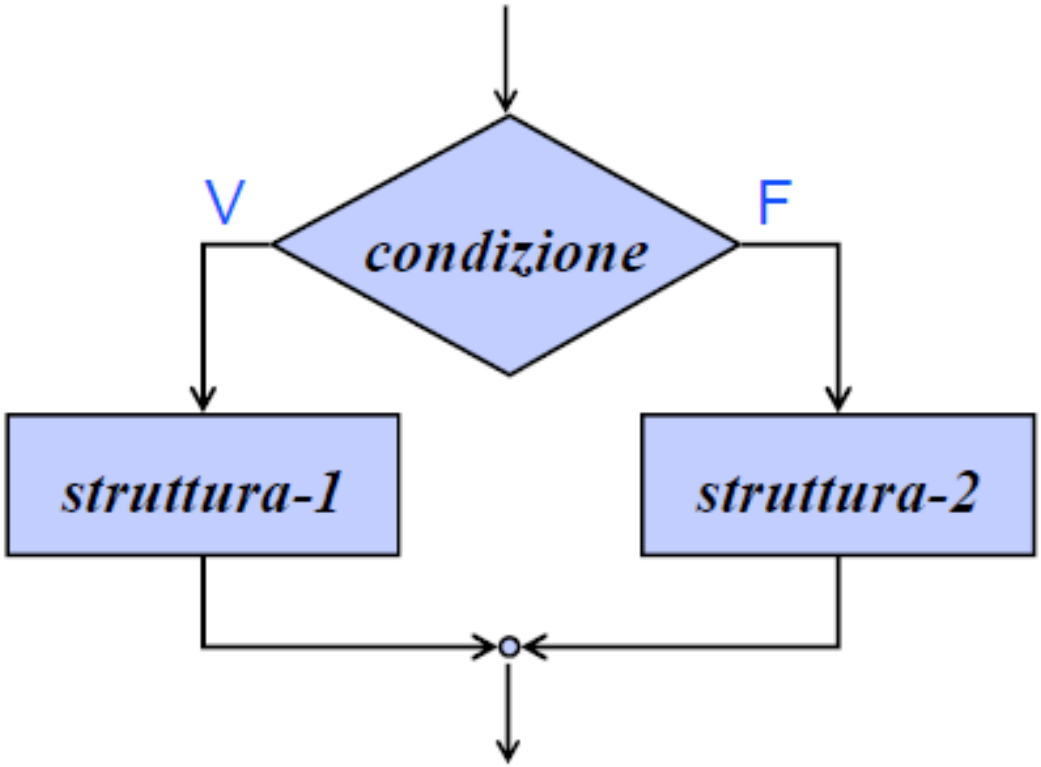
Un diagramma di flusso è detto *strutturato* se contiene solo un insieme predefinito di strutture:

- **sequenze**
- **decisioni**
  - **IF-THEN-ELSE**
  - **IF-THEN**
- **cicli**
  - **WHILE**
  - **REPEAT**

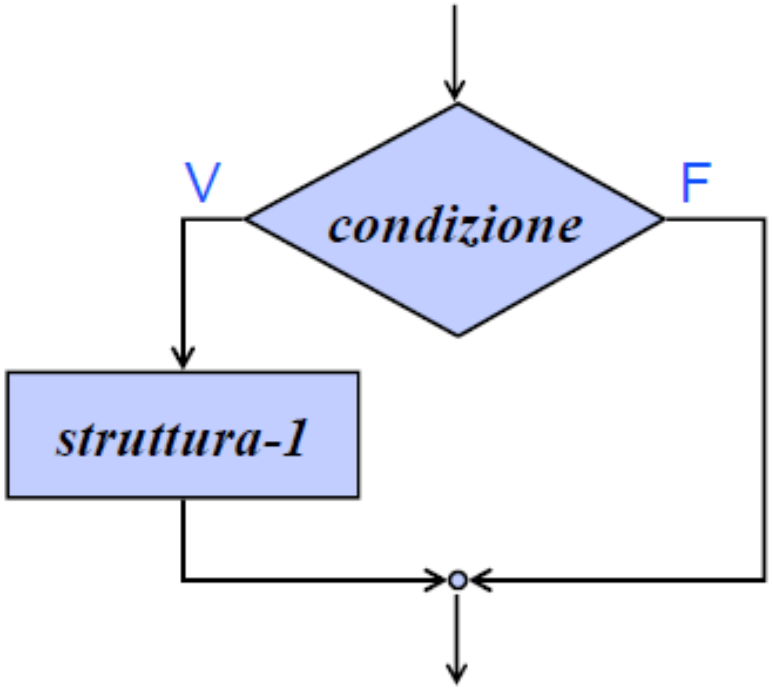
Teorema di Böhm - Jacopini

Qualunque diagramma di flusso è sempre trasformabile in un diagramma di flusso strutturato equivalente a quello dato.

If-Then-Else



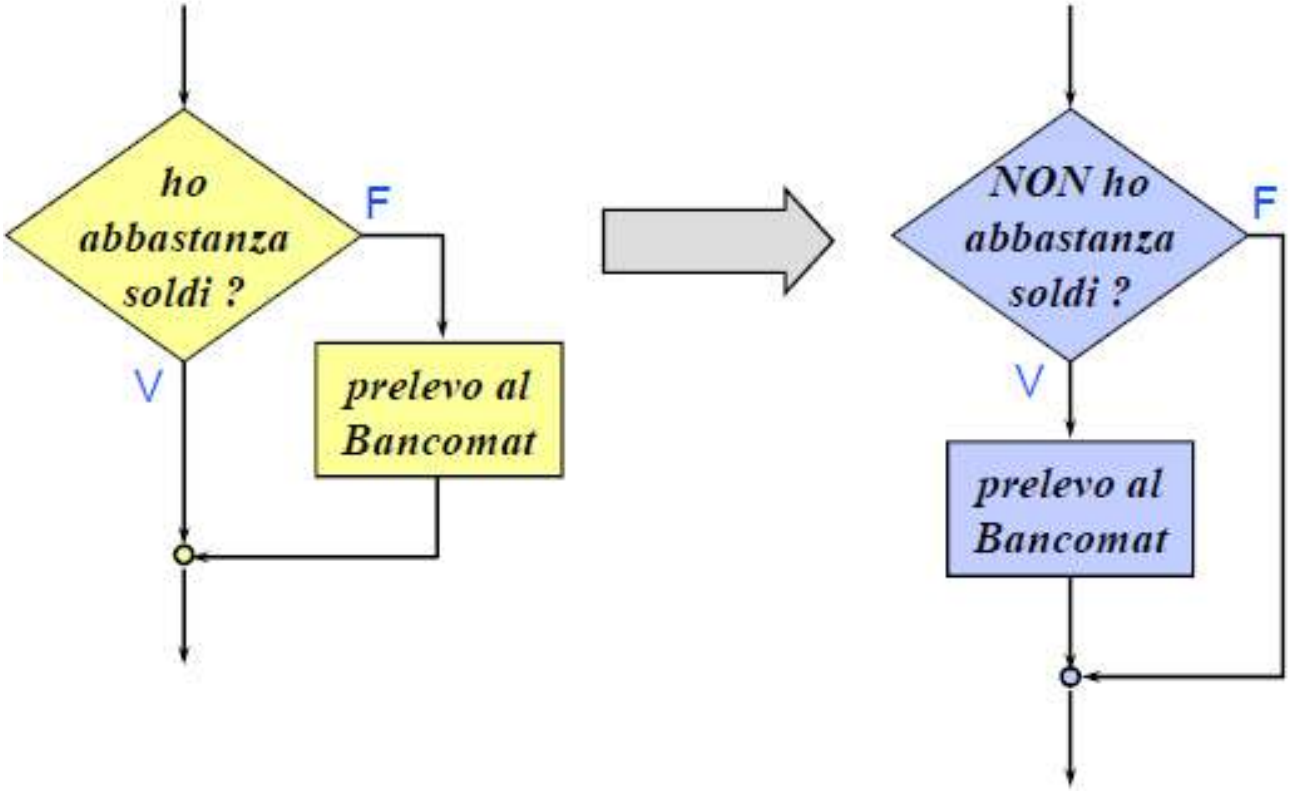
If-Then



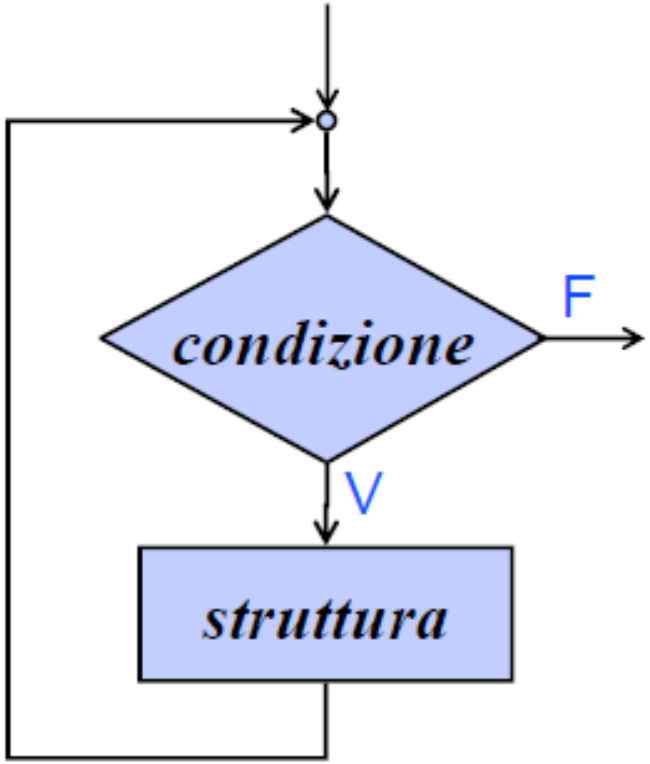
If-Else

- non esiste un blocco If-Else …
- … perché non è necessario!
- basta usare un blocco If-Then in cui la condizione sia stata invertita (negata)

If-Else (esempio)



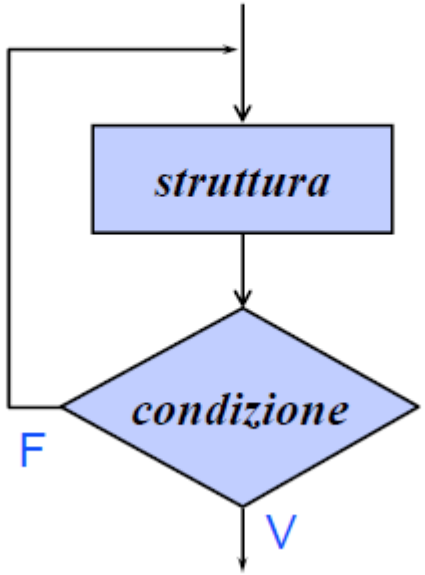
While-Do



While-Do

- la parte ciclica viene eseguita quando la condizione è vera
- se abbiamo un ciclo che viene eseguito quando la condizione è falsa, allora occorre trasformarlo in un While-Do mettendo la condizione negata
- un ciclo While-Do può essere eseguito zero o più volte
- viene eseguito zero volte quando la condizione è subito falsa

Repeat-Until



Repeat-Until

- **un ciclo Repeat-Until viene sempre eseguito almeno una volta**

Verifica di strutturazione

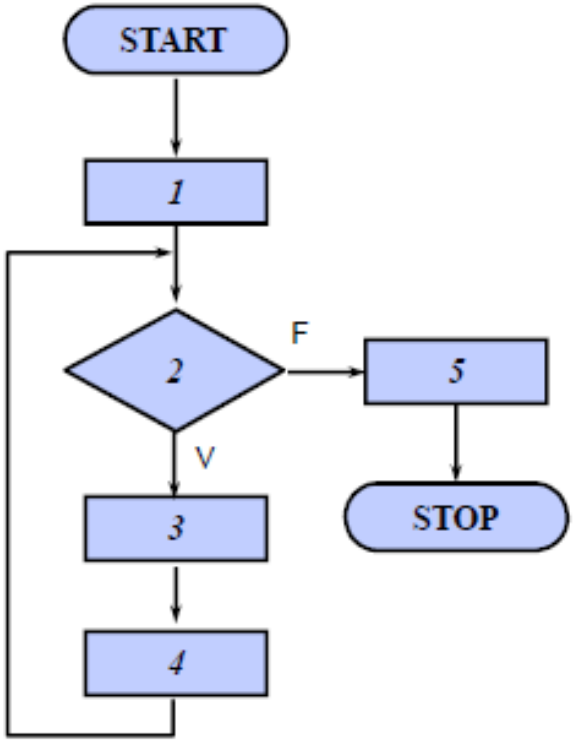
P1. etichettare ogni blocco

P2. sostituire ad ogni insieme strutturato un blocco avente come etichetta l'unione delle etichette dei blocchi che lo costituiscono

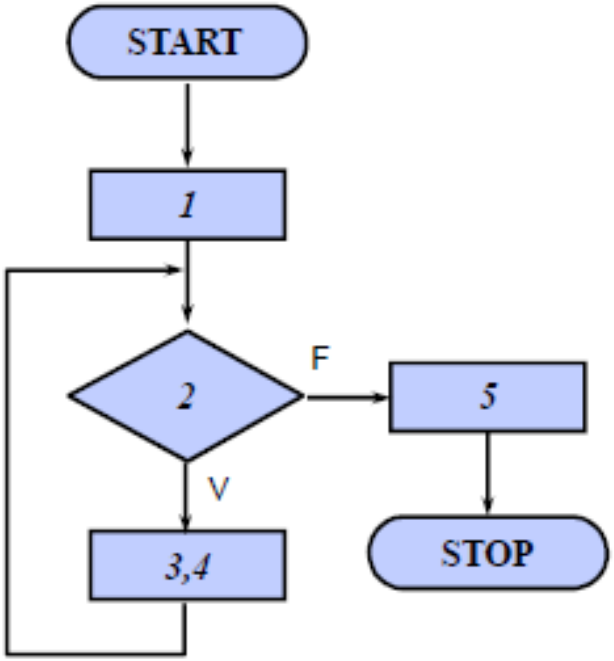
P3. se al passo P2 si è fatta almeno una sostituzione, ripetere il passo P2

P4. se alla fine si ottiene un diagramma lineare (una sequenza), allora il diagramma originale è strutturato

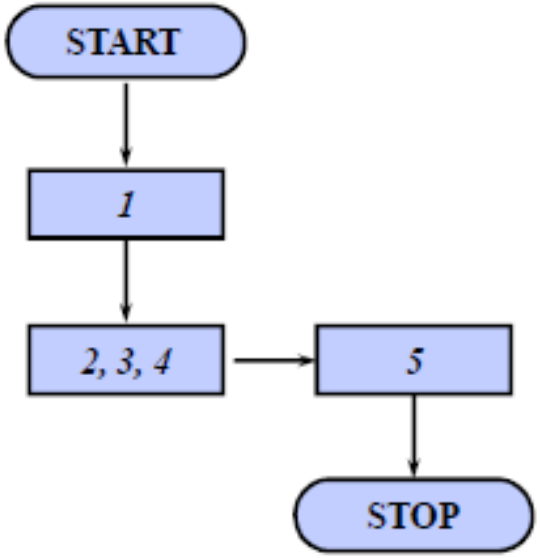
Esempio: diagramma strutturato



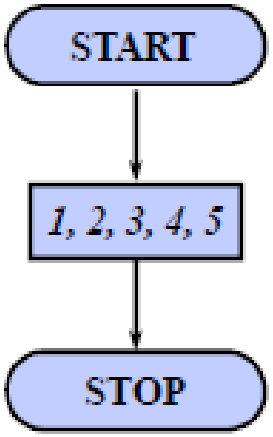
Esempio: diagramma strutturato



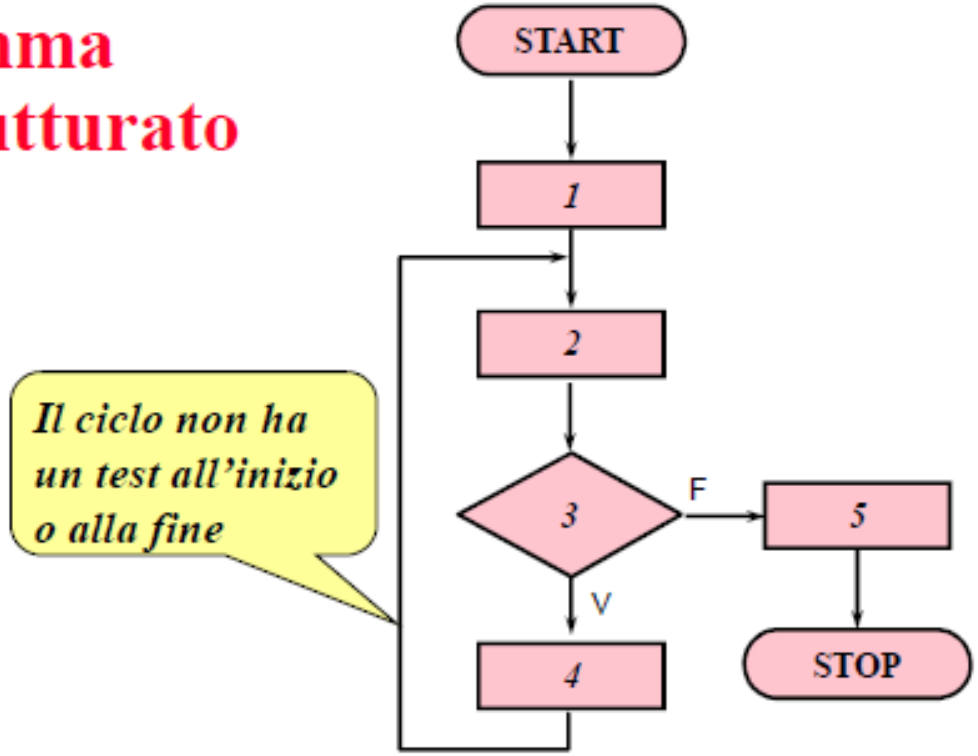
Esempio: diagramma strutturato



Esempio: diagramma strutturato



Esempio: diagramma non strutturato



PROBLEMA: DATI DUE NUMERI, INDIVIDUARE IL VALORE MASSIMO

SOLUZIONE:

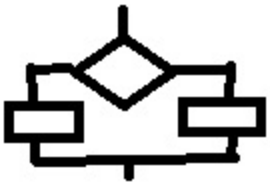
- CHIAMIAMO A e B LE CELLE DI MEMORIA IN CUI SONO CONTENUTI I DUE - VALORI NUMERICI
- LEGGIAMO IL VALORE MEMORIZZATO IN A
- LEGGIAMO IL VALORE MEMORIZZATO IN B
- SE A è MAGGIORE DI B ALLORA

    DIREMO CHE E' IL VALORE PRESENTE IN A IL NUMERO PIU' GRANDE

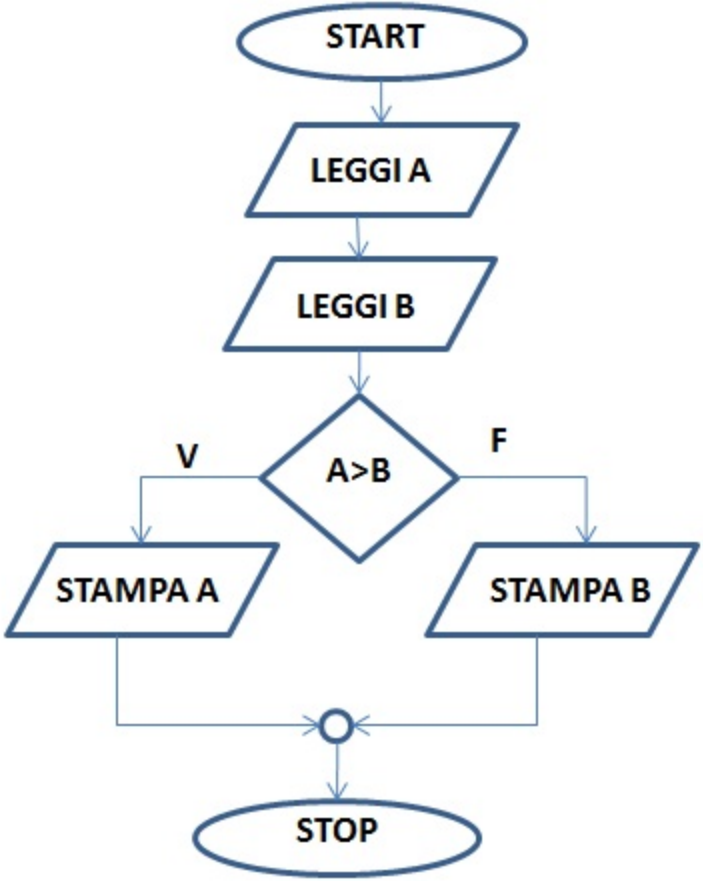
  ALTRIMENTI

    DIREMO CHE E' IL VALORE PRESENTE IN B IL NUMERO PIU' GRANDE

LEGGI o SCRIVI o STAMPA 

SE ALLORA ALTRIMENTI 

PROBLEMA: DATI DUE NUMERI, INDIVIDUARE IL VALORE MASSIMO

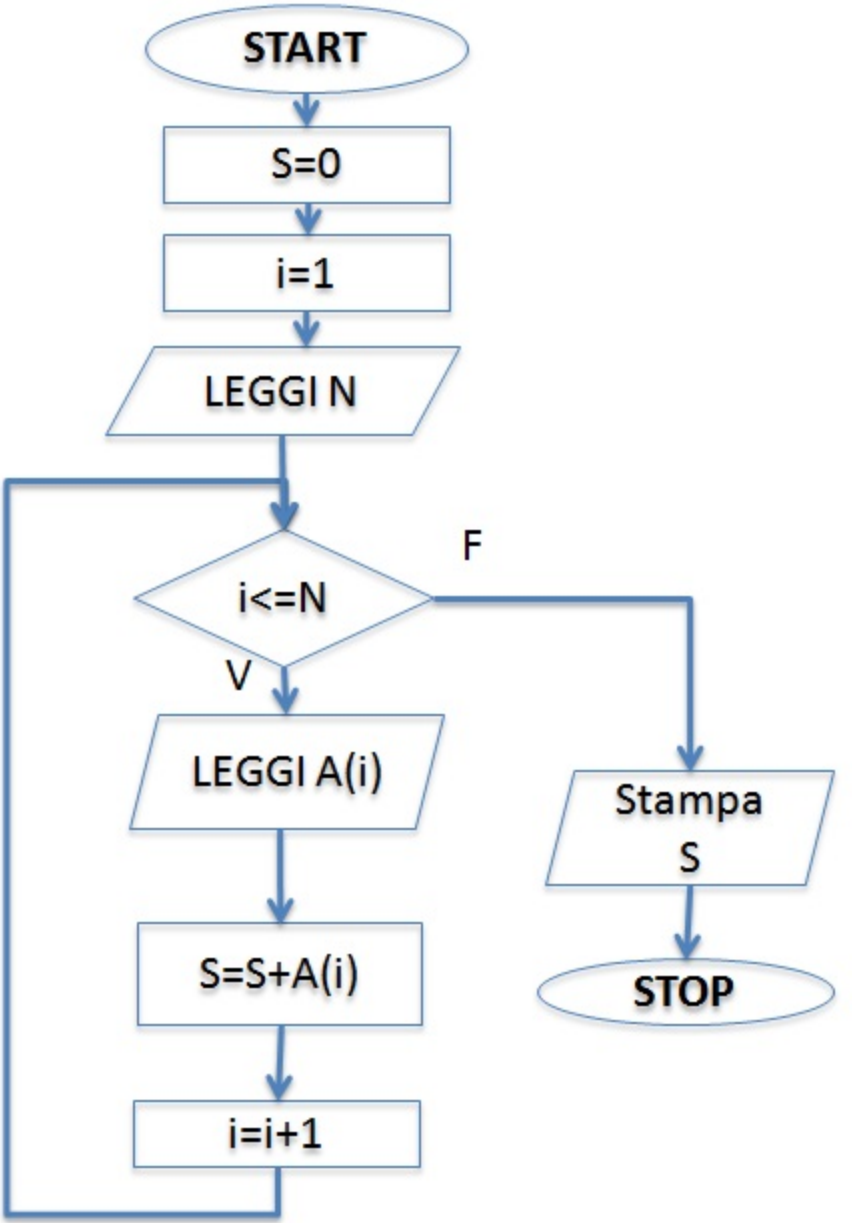


Supponiamo di avere il seguente array di N numeri. Ipotizziamo N=5

A(1)=2
A(2)=10
A(3)=4
A(4)=8
A(5)=1

Calcolare la somma

I	I<N	S
0		0
1	1<5 Si	S=S+A(1)= 0+2=2
2	2<5 SI	S=S+A(2)= 2+10=12
3	3<5 si	S=S+A(3)= 12+4=16
4	4<5 si	S=S+A(4)= 16+8=24
5	5<=5 si	S=S+A(5)= 24+1=25
6	6<5 no	Scrivi S=25



Un flow chat descrive i passaggi che devono essere eseguiti per risolvere un problema/trovare una soluzione.
In questo caso si suppone che **dato in input una sequenza di 10 numeri, si riesca a determinare il valore max.**
Chiamiamo A(i) la sequenza dei numero, dove "i" indica la posizione del numero da leggere.
Pertanto la nostra sequenza di numeri sarà A(1), A(2), A(3), A(4).....A(10).
Supponiamo che ad es. la ns sequenza sia
A(1)=8
A(2)=6
A(3)=12
A(4)=4
.....
A(10)=22
Vediamo come ragiona un calcolatore che applica il ns flow chat.
- INIZIO
- ASSEGNAMIAMO ALLA VARIABILE MAX il valore del ns primo valore dell'array, cioè di A(1).
Pertanto al primo passaggio **MAX=8**
-Assegniamo alla variabile i il valore 2
-- ci chiediamo A(i)>MAX? cioè A(2)>MAX? cioè 6>8?
- la risposta è NO
- assegniamo alla variabile i il valore di i+1.
Pertanto **i=2+1** dunque **i=3**
- ci chiediamo A(i)>MAX? cioè A(3)>MAX? cioè 12>8?
-- la risposta è SI pertanto MAX=A(3) cioè MAX=12
-...e così via seguendo l'iter del flow chart finchè non si giungerà alla FINE

i	max
2	8
3	12
4	12
5	...

Per risolvere i flow chart si consiglia di disegnare una tabella con tutti le variabili e ad ogni iterazione riportare il valore delle variabili a quell'istante