Corso di Programmazione

Problemi ed Algoritmi

II parte

Prof.ssa Teresa Roselli

teresa.roselli@uniba.it

Processo (d’esecuzione)

- Applicazione di un metodo solutivo ad una situazione problematica
  - Esecuzione delle operazioni da esso previste
- Può essere delegato ad un processore diverso dall’estensore del metodo solutivo
  - Essere umano
  - Sistema meccanico

Processo d'esecuzione
Requisiti per la Delega

- Algoritmo descritto perfettamente all'esecutore in termini di operazioni effettivamente eseguibili
  - Interpretazione non ambigua
  - Comportamento uniforme
- Esecutore meccanico (macchina)
  - Imprescindibilità dalle operazioni eseguibili
    - Operazioni basiche o Azioni primitive

Processi Sequenziali

- L’esecuzione di un’azione non può sovrapporsi all’esecuzione di un’altra
  - Possibilità di prevedere strade alternative da seguire al presentarsi di una certa condizione

Processi Sequenziali

- Per evitare incomprensioni, la descrizione del processo di esecuzione deve definire esattamente
  - Gli oggetti su cui operare
  - La sequenza esatta delle azioni da compiere
    - Prima operazione
    - Ultima operazione
  - La specifica dei controlli che determinano l'ordine di esecuzione delle azioni

indipendentemente dalla natura dell'esecutore

Rappresentazione di Algoritmi
Notazioni

	Grafico	**Strutturato**
Diagrammi di flusso	X	
Linguaggio Lineare		X
Alberi di Decomposizione	X	X
Grafi di Nassi - Schneidermann	X	X

N.B.: NON sono linguaggi di programmazione

Diagrammi di Flusso

- Il linguaggio dei diagrammi di flusso è un linguaggio grafico tipicamente utilizzato per trasmettere ad un esecutore umano la descrizione di un algoritmo o processo
  - Si parte dal punto iniziale
    - Si seguono i percorsi indicati, intraprendendo le azioni che via via si incontrano
    - In caso di percorsi alternativi, se ne sceglie uno a seconda della condizione specificata

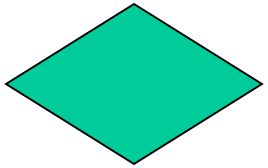
  fino al raggiungimento del punto finale

Diagrammi di Flusso
Elementi Costitutivi

- Operazioni
  - Calcolo (blocco azione)

  - Ingresso/Uscita

  - Decisione



- Controllo
  - Inizio/Fine (limiti del processo)

  - Flusso

  - Connessione

Diagrammi di Flusso
Definizione

E’ un grafo contenente:

- un blocco iniziale
- un blocco finale
- un numero finito di blocchi di azione
- un numero finito di blocchi di controllo

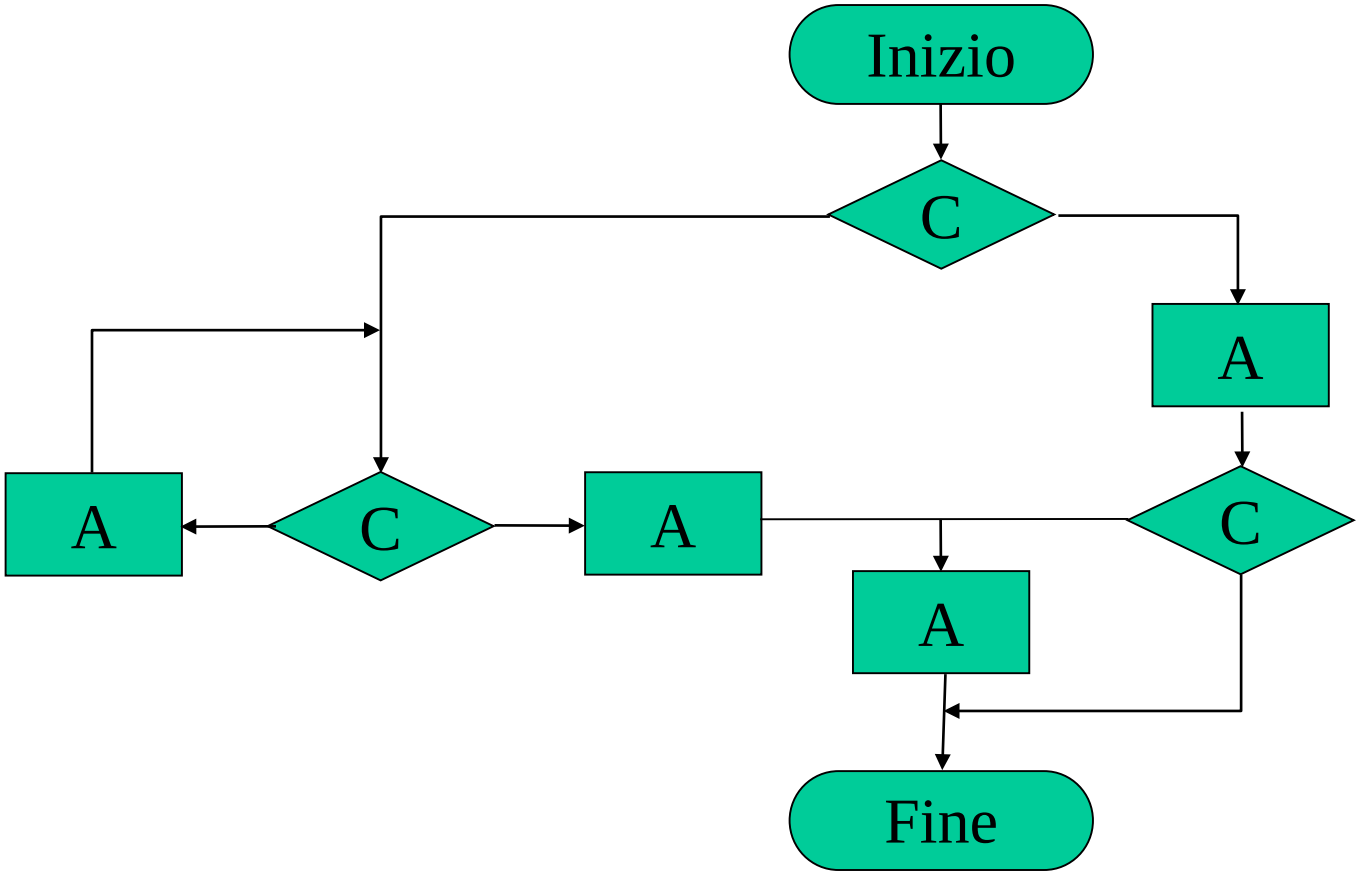
N.B. valgono le regole di costruzione seguenti

Diagrammi di Flusso
Regole di Costruzione

- *Un solo* blocco iniziale e *un solo* blocco finale
  - Ogni blocco è raggiungibile dal blocco iniziale
  - Il blocco finale è raggiungibile da ogni blocco
- I blocchi sono in numero finito
  - Ogni blocco di azione (calcolo o ingresso/uscita) ha *una* freccia entrante ed *una* uscente
  - Ogni blocco di decisione ha *una* freccia entrante e *due* uscenti
- Ogni freccia parte da un blocco e termina in un blocco o su un'altra freccia

Diagrammi di Flusso

Esempio



Diagrammi di Flusso

- La definizione data è costruttiva poiché è basata su regole che consentono di produrre diagrammi di flusso ma anche di riconoscerli

Diagrammi di Flusso
Punti di forza

- Grafici
  - Adatti agli esseri umani
  - Adatti a rappresentare processi sequenziali
  - Immediatamente visualizzabili
- Rispondono all’esigenza di divisione del lavoro
- Documento base per l’analisi organica
- Non ambigui
- Traducibili in vari linguaggi di programmazione

Diagrammi di Flusso
Punti di debolezza

- Spesso non entrano in una pagina
  - Difficili da seguire e modificare
- Non naturalmente strutturati
  - Spesso le modifiche portano a de-strutturazione
- Lontani dai linguaggi dei calcolatori
  - Possono rivelarsi errati in fase di programmazione

Diagrammi di Flusso Strutturati

- Esistono vari modi di connettere blocchi e frecce che rispettino la definizione di diagramma di flusso
- Gli schemi fondamentali o modelli di composizione fondamentali consentono di realizzare ***diagrammi di flusso strutturati***
  - Uso di soli diagrammi strutturati che corrispondono a configurazioni standard di blocchi elementari, comuni a molti processi della vita quotidiana
- Sviluppo per raffinamenti successivi
  - Ogni schema fondamentale ha un solo punto di ingresso e un solo punto di uscita
    - Sostituibile ad un blocco di azione
    - Nella sostituzione, si possono omettere i blocchi di inizio e fine dello schema che si sta inserendo

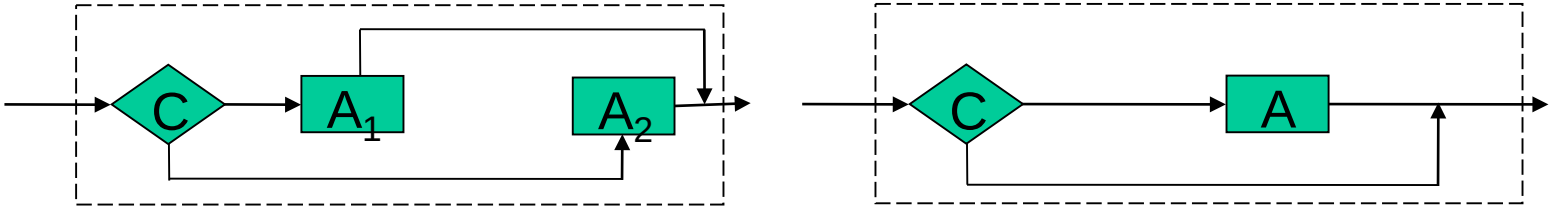
Diagrammi di Flusso Strutturati
Schemi fondamentali

- Sequenza
  - Concatenazione di azioni
- Selezione
  - Scelta di azioni alternative
    - Dipendenza da una condizione
- Iterazione
  - Ripetizione di una certa azione
    - Dati potenzialmente diversi
    - Dipendenza da una condizione

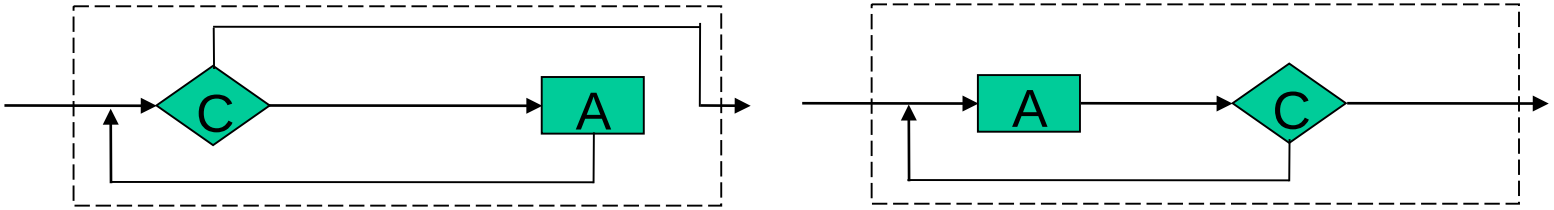
Diagrammi di Flusso Strutturati
Schemi fondamentali

- Sequenza

- Selezione



- Iterazione

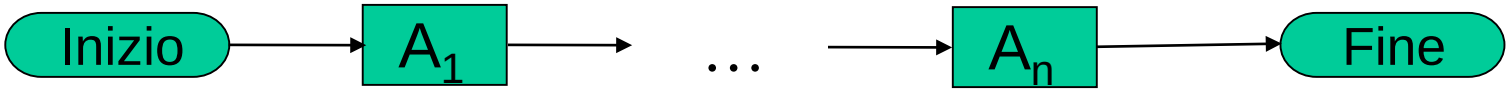


Diagrammi di Flusso Strutturati
Definizione

- **Base**: Dato un blocco di azione A,

è strutturato.

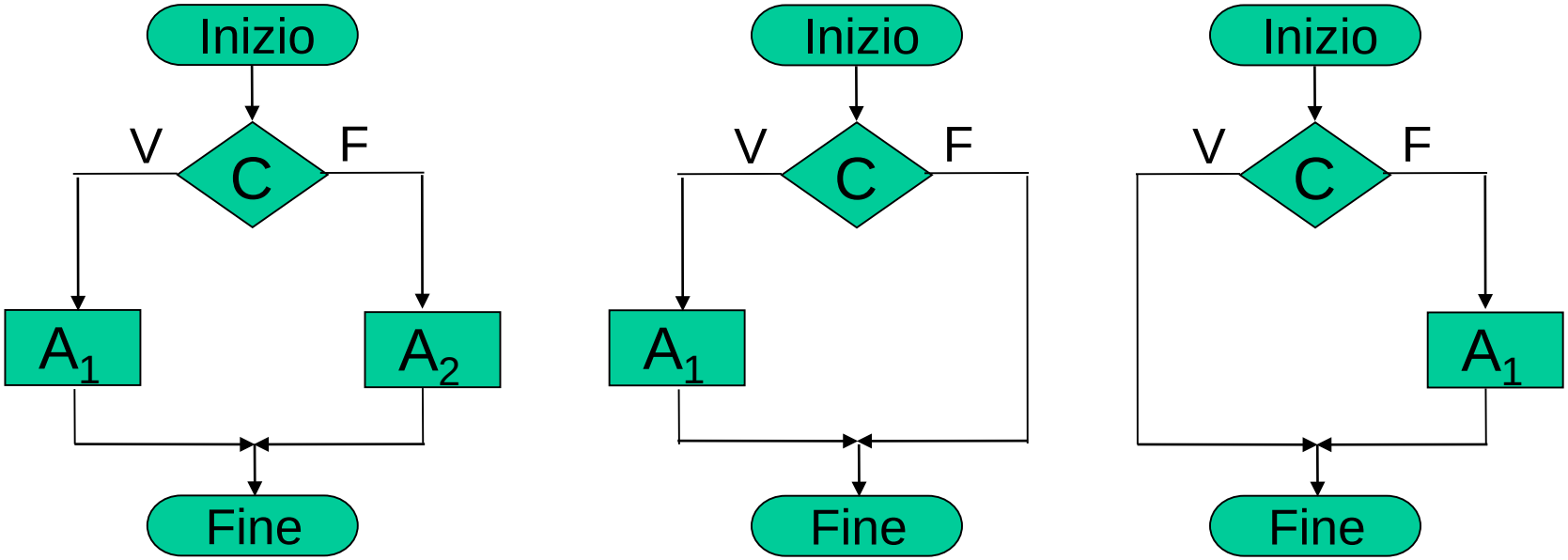
- **Sequenza**: Se A_1, …, A_n sono strutturati,



è strutturato

Diagrammi Strutturati

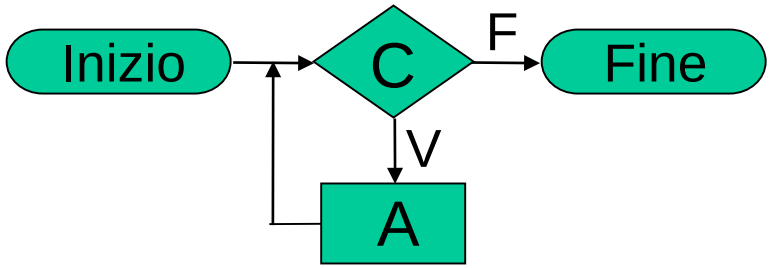
- **Selezione**: Se A_1 e A_2 sono strutturati,

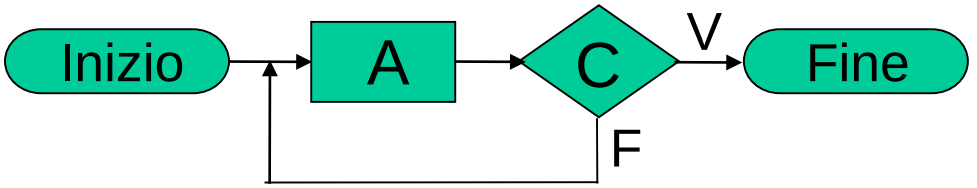


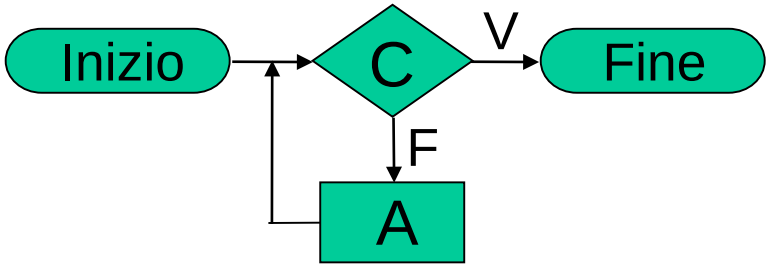
sono strutturati

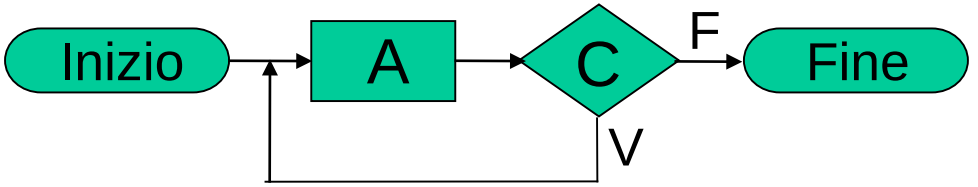
Diagrammi Strutturati

- **Iterazione**: Se A è strutturato,









sono strutturati

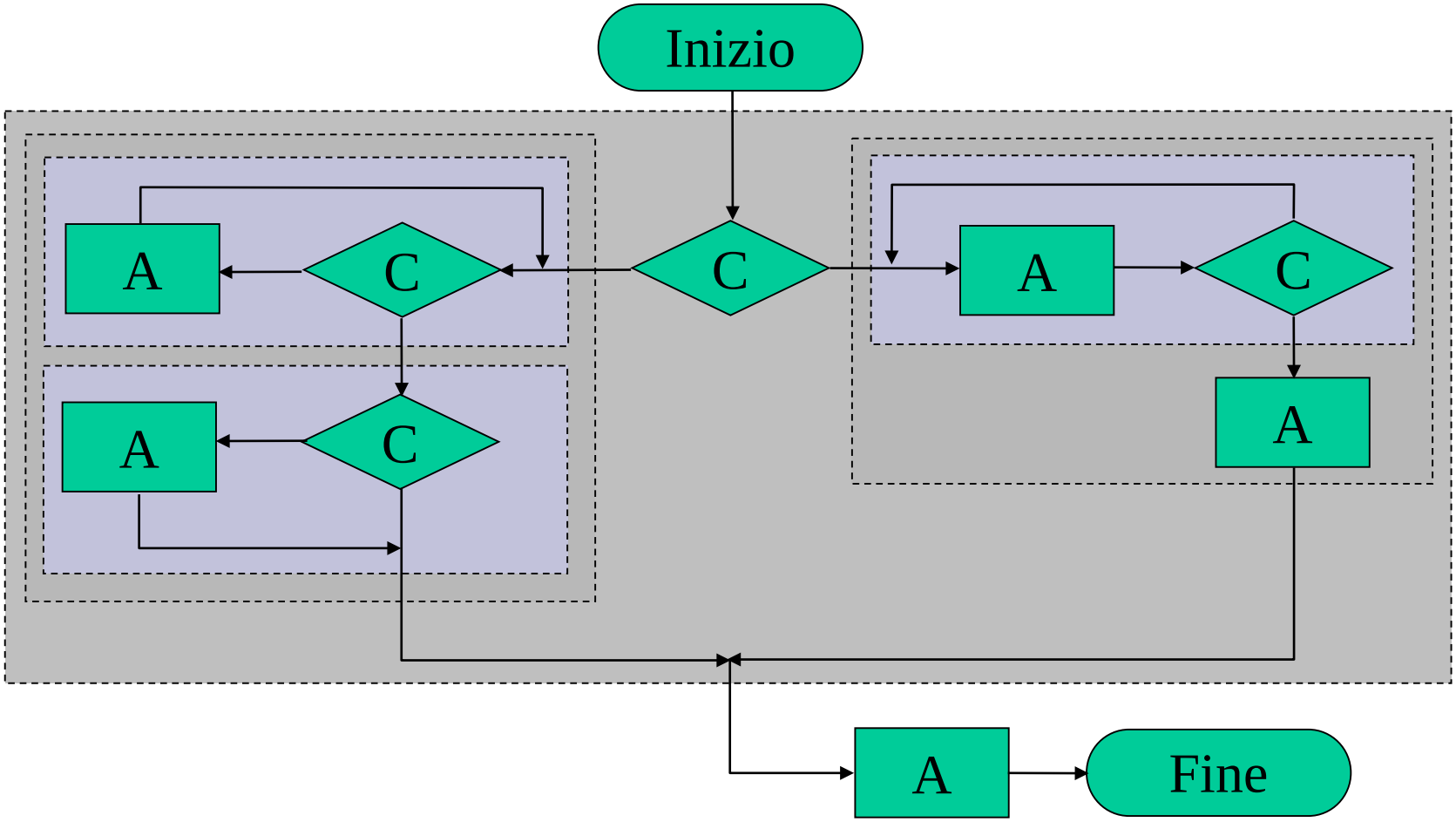
Diagrammi Strutturati

- Nessun altro diagramma è strutturato

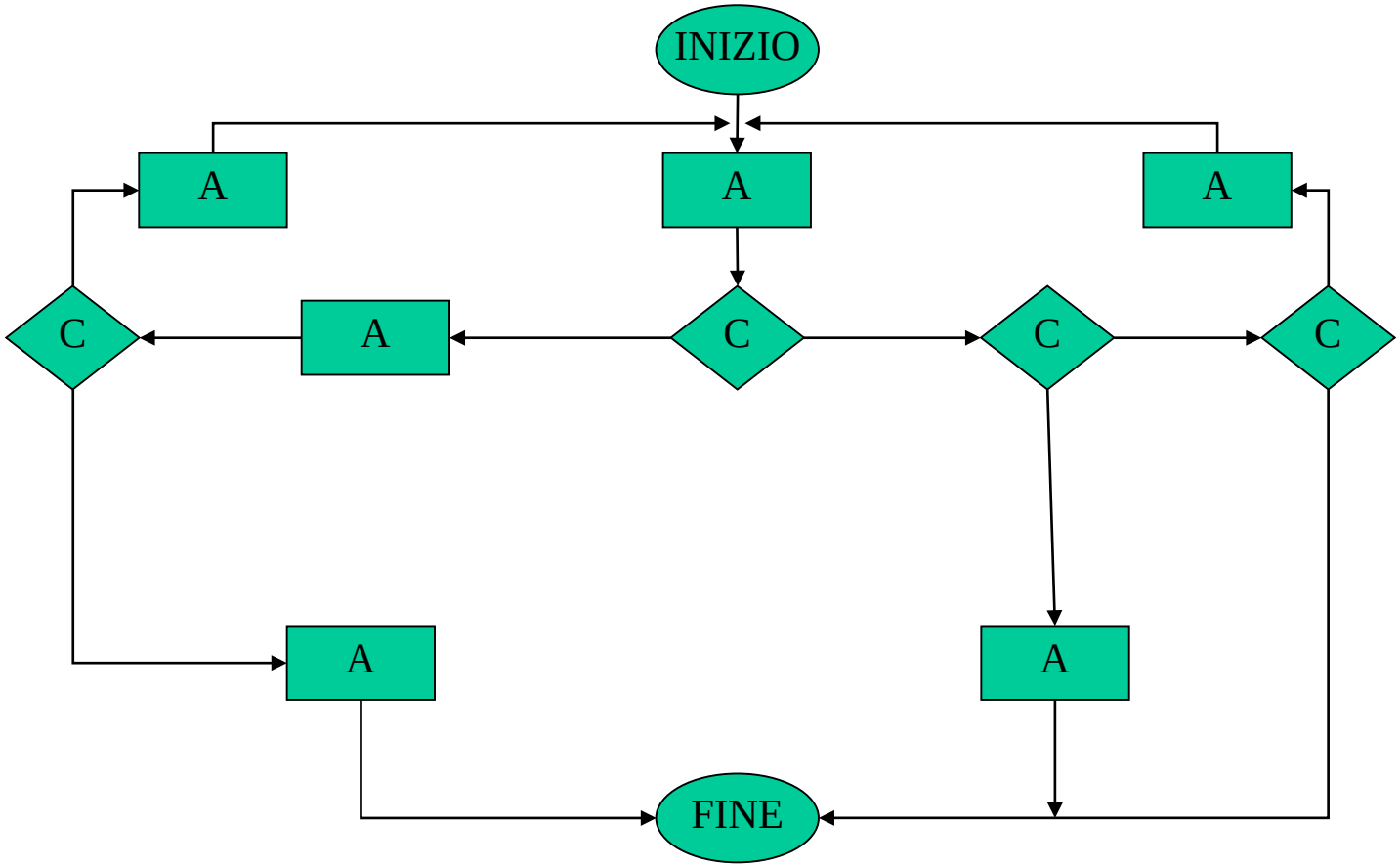
- Note:
  - Definizione ricorsiva

Diagrammi Strutturati

Esempio



Diagrammi non Strutturati
Esempio



Teorema di Böhm-Jacopini

- Dato un processo *P* e un diagramma che lo descrive, è sempre possibile determinare un processo *Q*, equivalente a *P*, che sia descrivibile tramite un **diagramma di flusso strutturato**
- Due processi applicati allo stesso insieme di dati si dicono **equivalenti** se producono lo stesso effetto
- Due processi equivalenti applicati agli stessi dati di ingresso o non terminano o terminano entrambi producendo gli stessi dati di uscita
- Un processo o metodo solutivo può essere sempre descritto tramite diagrammi strutturati

VIETATO
l’uso di istruzioni di salto

- Non necessarie
  - Teorema di Böhm-Jacopini
- Potenzialmente dannose
  - Difficoltà a seguire il flusso del controllo
    - Scarsa modificabilità
    - Interazioni impreviste
  - Goto statement considered harmful [Dijkstra, 68]

Linguaggio Lineare

- Atto alla descrizione di algoritmi
  - Costrutti linguistici non ambigui
- Usa esclusivamente schemi strutturati
- Simile ad un linguaggio di programmazione
  - Sparks [Horowitz, 1978]
    - Corrispondente italiano

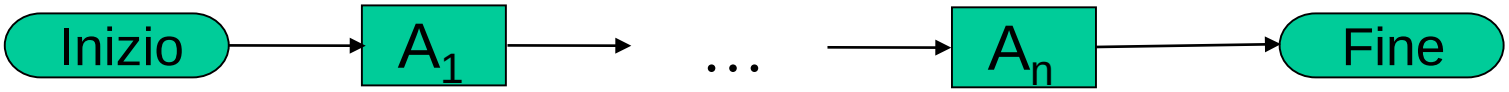
Linguaggio Lineare
Sequenza

- Costrutto base:

  begin A **end**

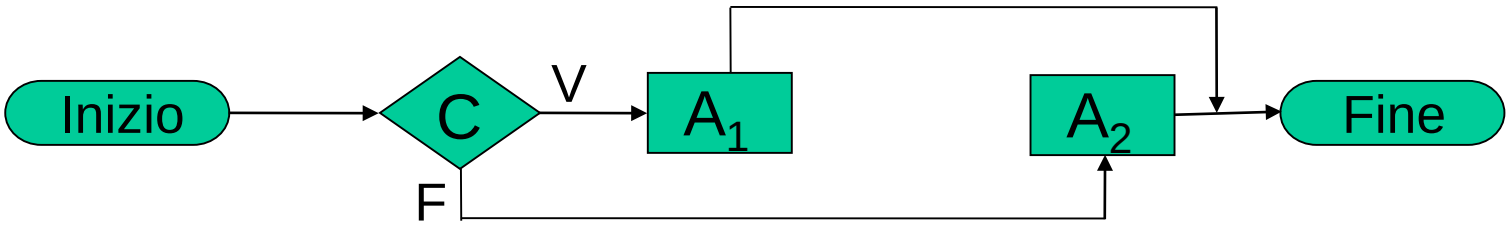
- Inoltre, ogni blocco di azione si può sostituire con uno dei seguenti costrutti

begin A_1 **;** … **;** A_n **end**

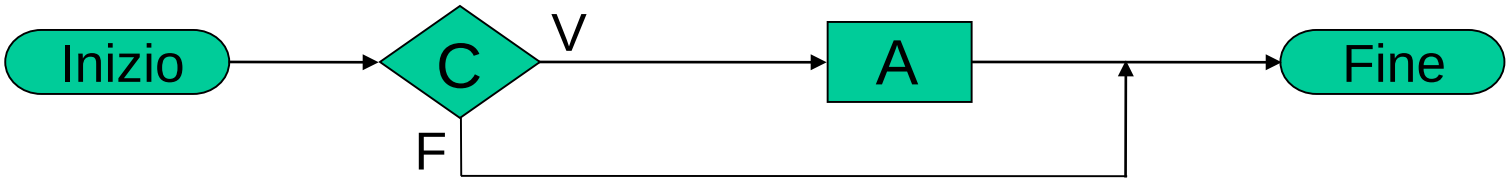


Linguaggio Lineare
Selezione

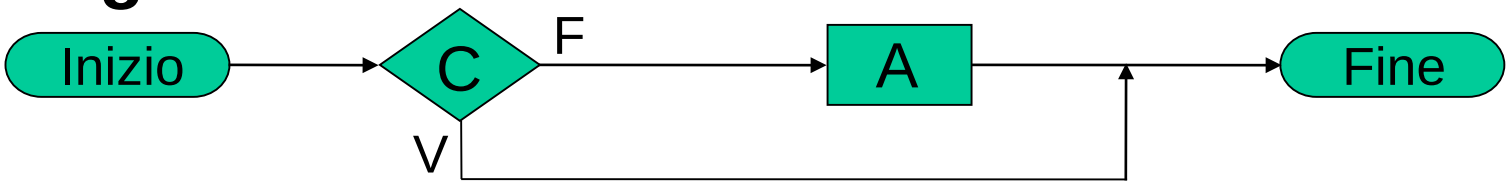
begin **if** C **then** A_1 **else** A_2 **end**



begin **if** C **then** A **end**

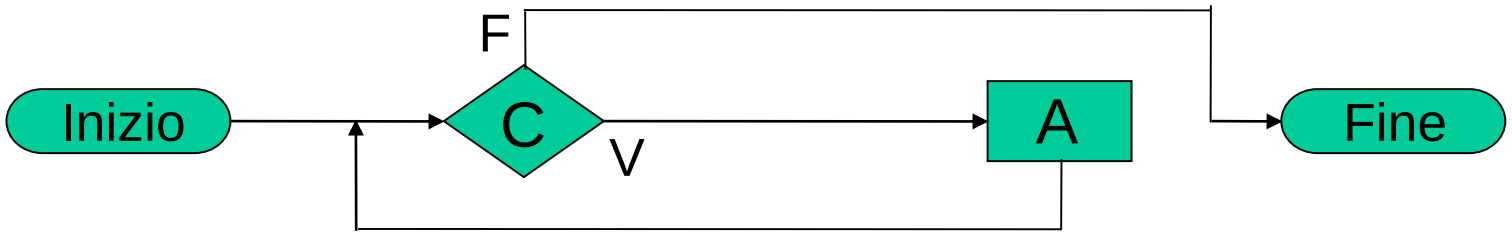


begin **if not** C **then** A **end**

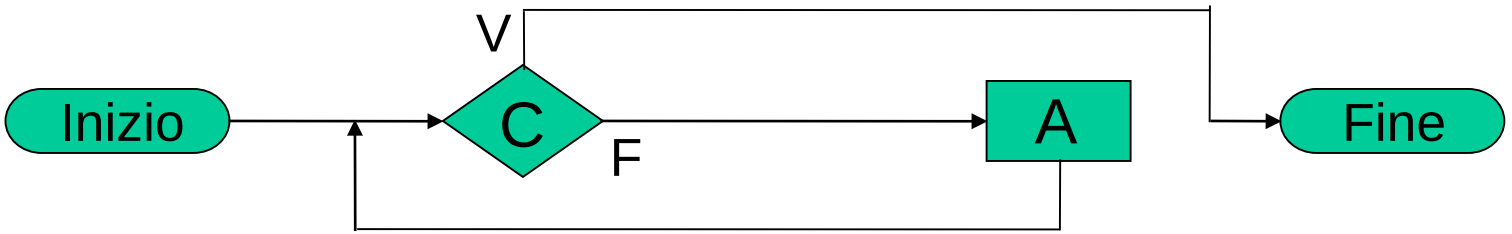


Linguaggio Lineare
Iterazione (while…do)

begin **while** C **do** A **end**

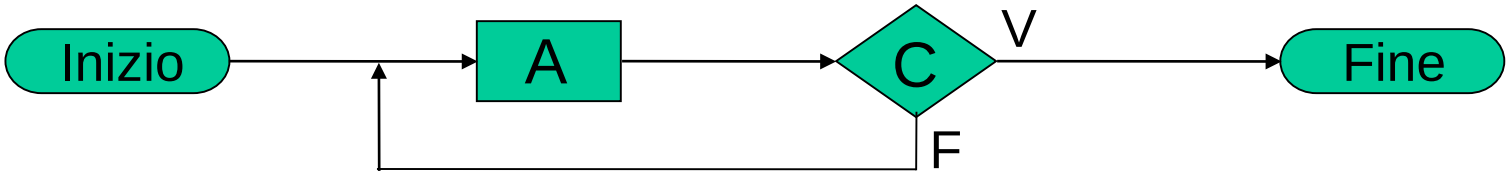


begin **while not** C **do** A **end**

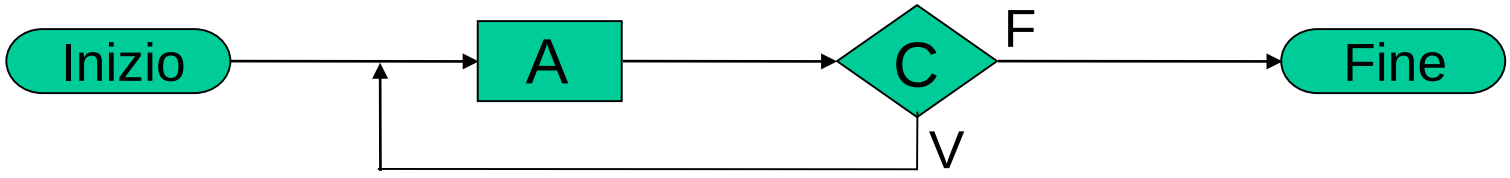


Linguaggio Lineare
Iterazione (repeat…until)

begin **repeat** A **until** C **end**

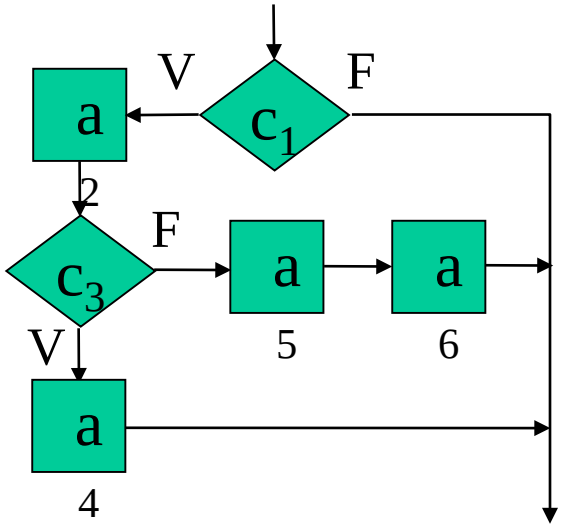


begin **repeat** A **until not** C **end**

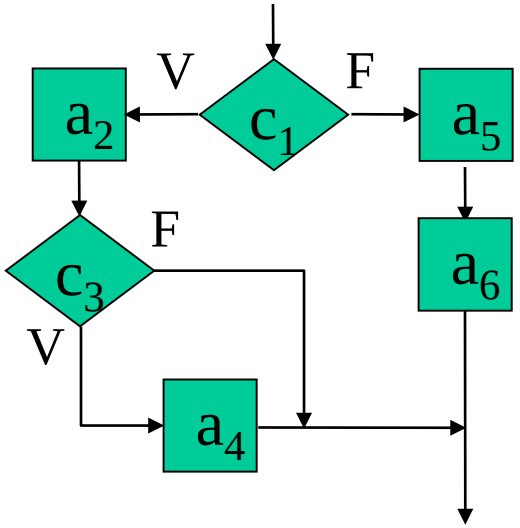


Linguaggio Lineare
Ambiguità

- **if** c_1 **then** a_2; **if** c_3 **then** a_4 **else** a_5; a_6



oppure ?



- Uso dell'indentazione
  - Aiuta ma non risolve

Risoluzione delle Ambiguità
Convenzioni aggiuntive

- Ogni descrizione di un sottoprocesso che sia composizione in sequenza di descrizioni di azioni elementari o sottoprocessi deve essere racchiuso tra le parole **begin – end**
  - Vale, in particolare, per la descrizione di un sottoprocesso (blocco di azioni) che segue le parole chiave
    - **then**
    - **else**
    - **while**

    quando non è un'azione basica

Risoluzione delle Ambiguità
Convenzioni aggiuntive

- Aggiungere i seguenti delimitatori di istruzione
  - Selezione: **endif**
  - Iterazione di tipo *while*: **endwhile**
    - Non necessario per l'iterazione di tipo *repeat*
      - È già presente la clausola *until* come delimitatore

Schemi ridondanti

- Doppio costrutto iterativo
  - Ciascuno dei costrutti **while** e **repeat** è ridondante una volta che sia disponibile l'altro
- Iterazione limitata
  - Basata sulla variazione di un indice di cui sono noti il valore iniziale, il valore finale e l'incremento o passo
- Selezione multipla
  - Basata sul partizionamento dei valori risultanti da una espressione in diverse classi di equivalenza rispetto all'azione da intraprendere

Schemi Ridondanti

```
while C do
    A
endwhile
```

è equivalente a

```
if C then
    repeat
     A
    until not C
endif
```

```
repeat
    A
until C
```

è equivalente a

```
A
while not C do
    A
endwhile
```

Schemi Ridondanti

do varying i
  from expr1 **to** expr2
   A
repeat

i ← expr1
while i ≤ expr2 **do**
  A**;**
  i ← i + 1
endwhile

Se l'incremento o passo non è specificato allora vale 1

Schemi Ridondanti

case espr **of**
    $lista_1$ **:** A_1**;**
    $lista_2$ **:** A_2**;**
    …
    $lista_n$ **:** A_n**;**
else
    A_0
endcase

if espr in $lista_1$
    then A_1
else if espr in $lista_2$
    then A_2
…
else if espr in $lista_n$
    then A_n
else A_0
endif

Schemi ridondanti
Esempio

- Iterazione limitata
  - Quadrato dei primi *n* numeri interi

per i **che va**
  da 0 **a** n
  stampa i * i
ripeti

- Selezione multipla
  - Numero di giorni in un mese

nel caso che mese **sia**
  11,4,6,9 **:** giorni ← 30
  2 **:** giorni ← 28
altrimenti
  giorni ← 31
finecasi

Massimo Comune Divisore

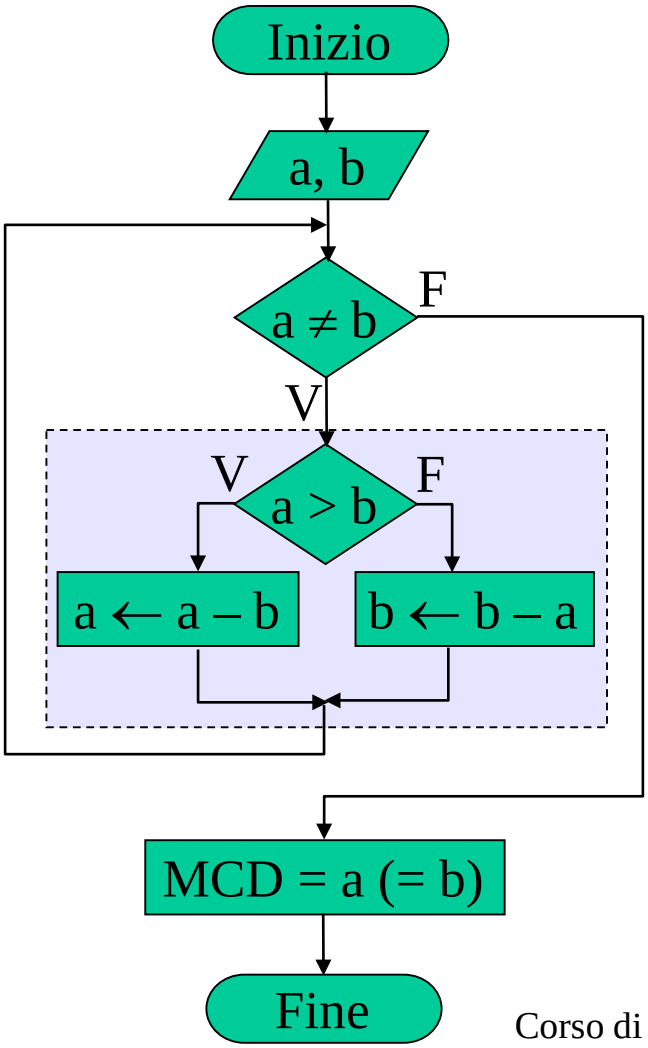
- Il massimo comune divisore può essere calcolato, in linea di principio, determinando la scomposizione in fattori primi dei due numeri dati e moltiplicando i fattori comuni, considerati una sola volta con il loro minimo esponente. Per esempio, per calcolare il MCD(18,84) si scompongono dapprima i due numeri in fattori primi, ottenendo 18 = 2·32 e 84 = 22·3·7, e poi si considerano i fattori comuni ai due numeri, 2 e 3: entrambi compaiono con esponente minimo uguale a 1, e quindi si ottiene che MCD(18,84)=6. Non trovando fattori primi comuni, il MCD è 1, così ad esempio MCD(242,375)=1.

- Un metodo molto più efficiente è fornito dall'algoritmo di Euclide: si divide 84 per 18 ottenendo un quoziente di 4 e un resto di 12. Poi si divide 18 per 12 ottenendo un quoziente di 1 e un resto di 6. Infine si divide 12 per 6 ottenendo un resto di 0, il che significa che 6 è il massimo comun divisore.

Esempio
Algoritmo Euclideo per il MCD

- ***Considera*** la coppia di numeri dati
- **Fintantoché(mentre)** i numeri sono diversi **ripeti**
  - **Se** il primo numero è minore del secondo **allora**
    - Scambiali
  - **Sottrai** il secondo dal primo
  - **Rimpiazza** i due numeri col sottraendo e con la differenza, rispettivamente
- Il ***risultato*** è il valore ottenuto

Esempio
Algoritmo Euclideo per il MCD




```
begin
    leggi a, b
    while (a ≠ b) do
     if (a > b) then
          a ← a – b
     else
          b ← b – a
    endif
    MCD ← a
end
```

a	b
84	18
66	18
48	18
30	18
12	18
12	6
6	6

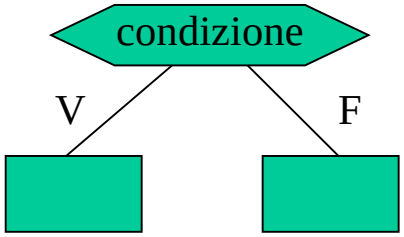
MCD = 6

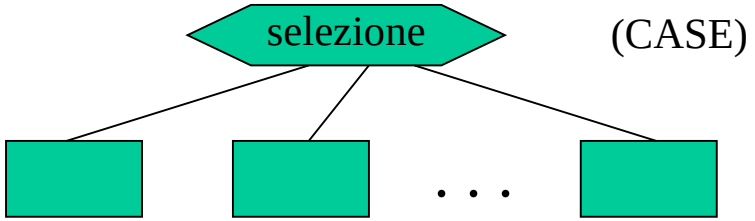
Alberi di Decomposizione

- Rappresenta tramite la relazione padre-figlio la scomposizione di una operazione in operazioni più semplici
  - Più strutturata
  - Consente un'analisi dall'alto verso il basso
    - Adatta alla scomposizione per raffinamenti successivi

Alberi di Decomposizione

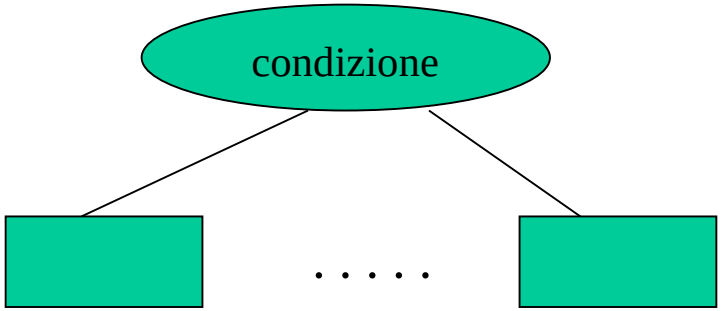
- Sequenza
  - operazioni su uno stesso livello da sinistra verso destra
- Selezione
  - blocco di condizione che si diparte in più strade





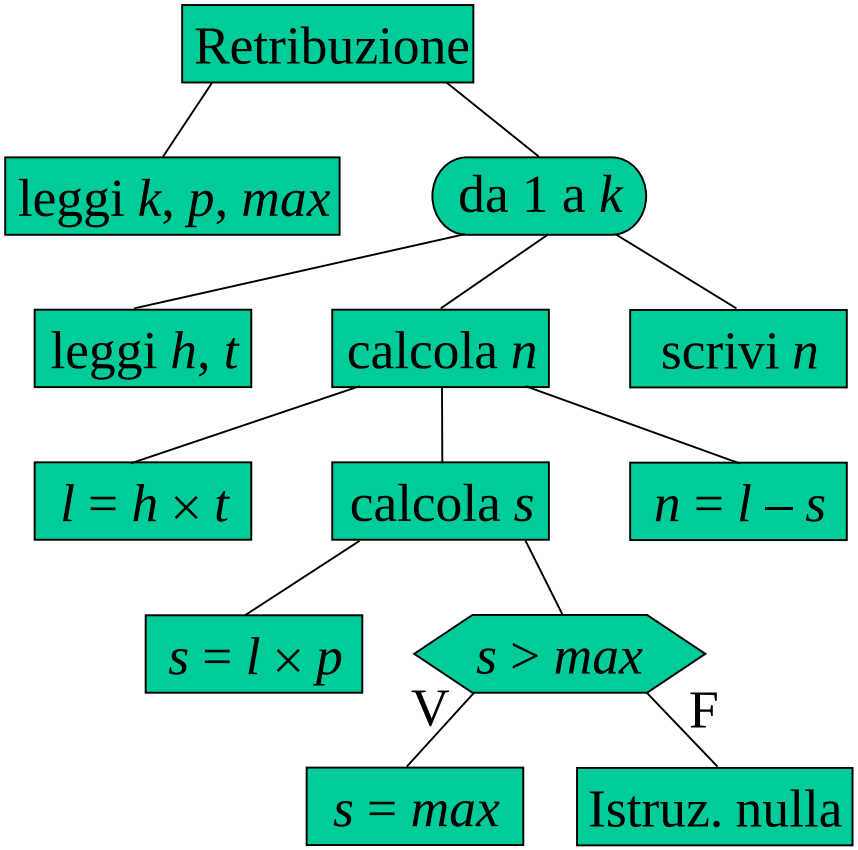
Alberi di Decomposizione

- Iterazione
  - blocco di condizione che controlla la terminazione dei nodi figli



Alberi di Decomposizione
Esempio

- Calcolo retribuzione al netto delle trattenute per *k* individui
  - *t* ore lavorate
  - *h* retribuzione oraria
  - *p* percentuale trattenute su retribuzione base
  - *max* tetto trattenute
  - *n* retribuzione netta
  - *l* retribuzione lorda
  - *s* trattenute calcolate

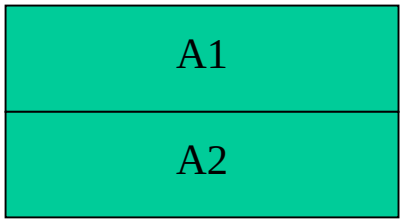


Grafi di Nassi-Schneidermann

- Uniscono il vantaggio di una rappresentazione grafica con quello di poter rappresentare schematicamente metodi strutturati

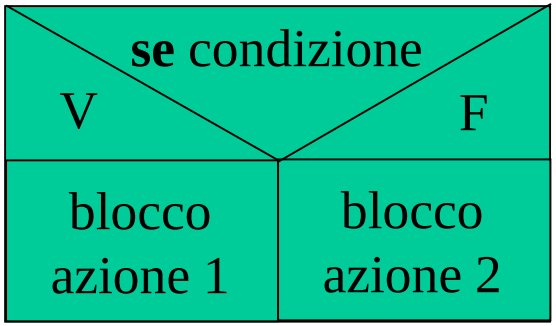
SCHEMI

- Sequenza

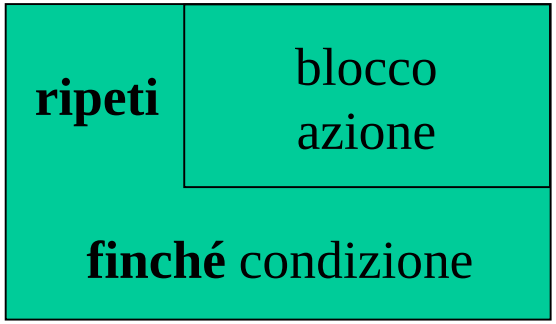


Grafi di Nassi-Schneidermann
Schemi

- Selezione

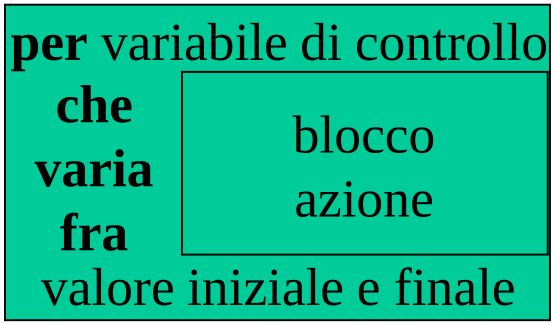


- Iterazione repeat



- Iterazione while

- Iterazione limitata



Grafi di Nassi-Schneidermann
Esempio

- Calcolo retribuzione al netto delle trattenute per *k* individui
  - *t* ore lavorate
  - *h* retribuzione oraria
  - *p* percentuale trattenute su retribuzione base
  - *max* tetto trattenute
  - *n* retribuzione netta
  - *l* retribuzione lorda
  - *s* trattenute calcolate