Rappresentazione degli Algoritmi e Programmazione Strutturata

Sommario

- Diagrammi di Flusso
- Programmazione Strutturata
- Linguaggio Lineare

Rappresentazione degli algoritmi

- Negli esempi, gli algoritmi sono rappresentati in un linguaggio simile a quello naturale
- Vantaggi
  - Intuitività
  - Facilità di scrittura
- Svantaggi
  - Ambiguità
  - Ridondanza
  - Scarso rigore

Diagrammi di Flusso

- 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

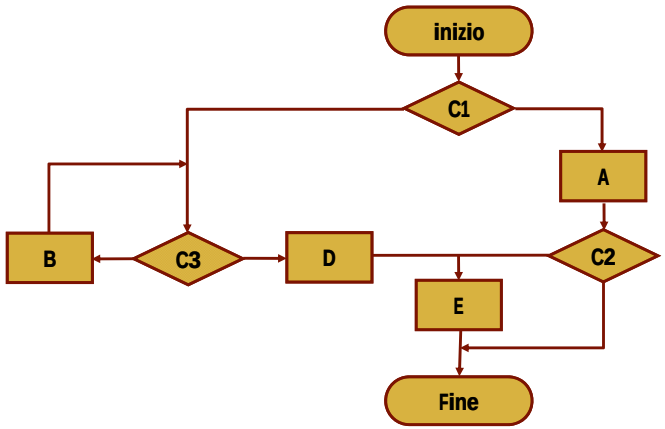
Diagrammi di Flusso
Elementi Costitutivi

- Operazioni
  - Calcolo
  - Ingresso/Uscita
  - Decisione
- Controllo
  - Inizio/Fine
  - Flusso
  - Connessione

Diagrammi di Flusso
Regole di Costruzione

- Un solo blocco iniziale e un solo blocco finale
  - Ogni blocco è raggiungibile da quello 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:
Vantaggi

- 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
- Poco ambigui

Diagrammi di Flusso: Svantaggi

- 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

Programmazione Strutturata

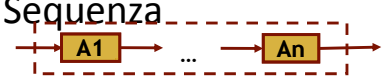
- Uso di schemi fondamentali
  - Uso di soli diagrammi strutturati
    - Configurazioni standard di blocchi elementari, comuni a molti processi della vita quotidiana
- Sviluppo per raffinamenti successivi
  - Ogni schema fondamentale ha un solo punto di ingresso ed un solo punto di uscita
    - Sostituibile ad un blocco di azione
    - Nella sostituzione, si possono omettere i blocchi di inizio e fine dello schema che si sta inserendo

Programmazione Strutturata: Schemi fondamentali (1)

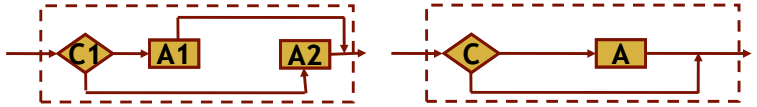
- Sequenza
  - Concatenazione di azioni
- Selezione
  - Scelta di azioni alternative
    - Dipendenza da una condizione
- Iterazione
  - Ripetizione di una certa azione
    - Dati potenzialmente diversi
    - Dipendenza da una condizione

Programmazione Strutturata: Schemi fondamentali (2)

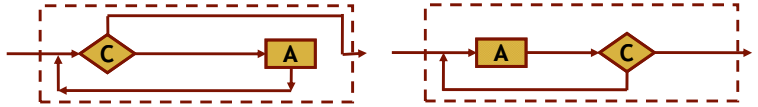
- Sequenza



- Selezione



- Iterazione

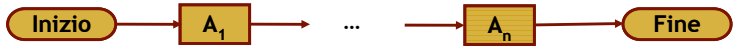


Diagrammi Strutturati (1)

- Base: Dato un blocco di azione A,

è strutturato.

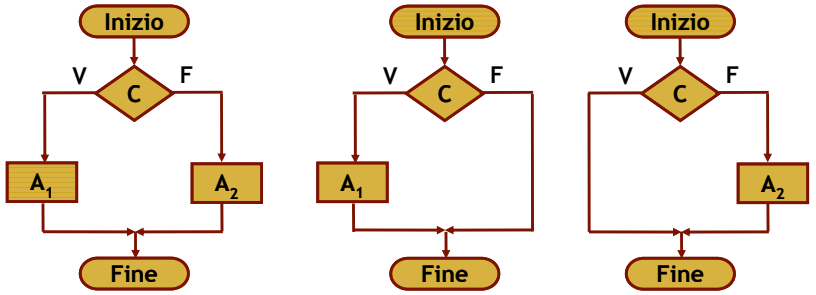
- Sequenza: Se A1, …, An sono strutturati,



è strutturato

Diagrammi Strutturati (2)

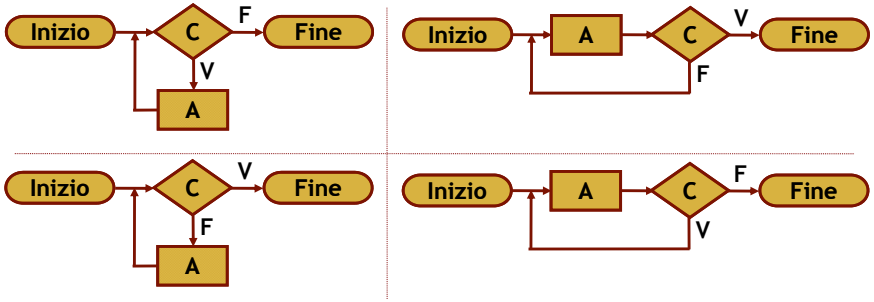
- Selezione: Se A1 e A2 sono strutturati,



sono strutturati

Diagrammi Strutturati (3)

- Iterazione: Se A è strutturato allora...

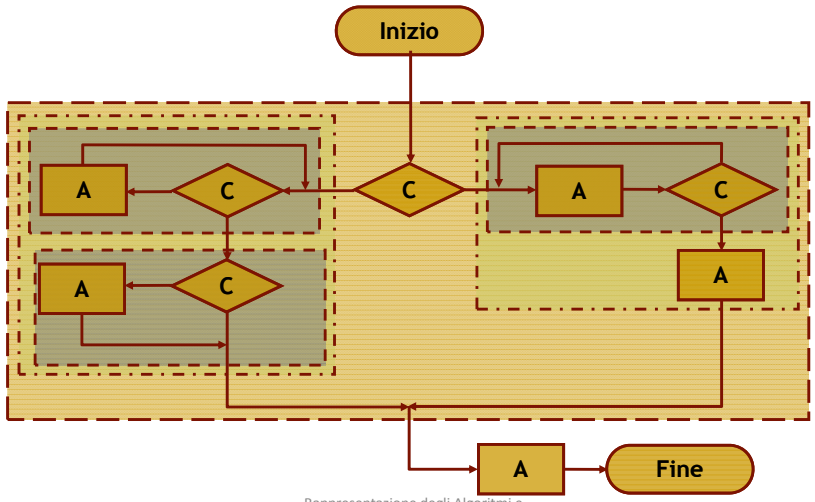


sono diagrammi strutturati

Diagrammi Strutturati (4)

- Nessun altro diagramma è strutturato
- Note:
  - La definizione è ricorsiva

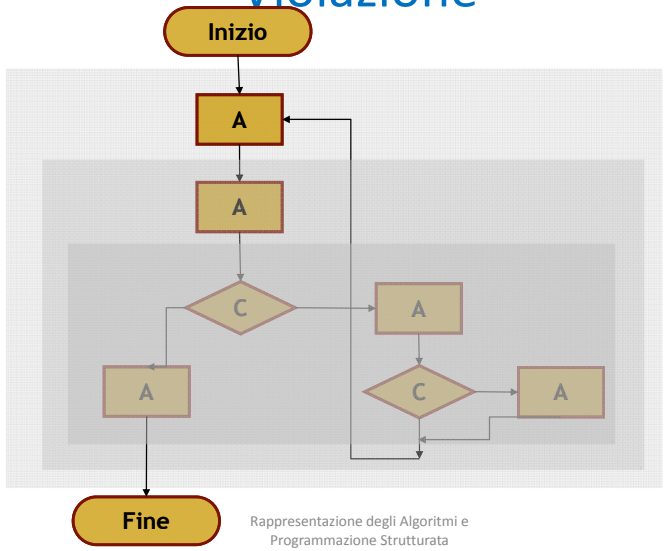
Diagrammi Strutturati
Esempio



Teorema di Böhm-Jacopini

- Dato un processo sequenziale P e un diagramma che lo descrive, è sempre possibile determinare un processo Q, equivalente a P, che sia descrivibile mediante un diagramma strutturato
  - Si definiscono equivalenti due processi che producono lo stesso effetto
- Un processo o metodo solutivo può essere sempre descritto tramite diagrammi strutturati

Diagmammi Strutturati
Violazione



DIVIETO
dell'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, 1968]

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:

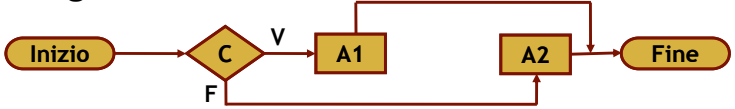
begin A end

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

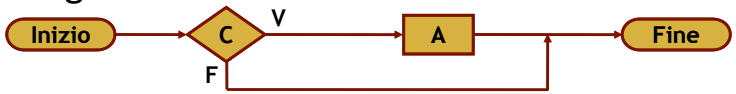
begin A1; … ; An end

Linguaggio Lineare Selezione

- begin if C then A1 else A2 end



- begin if C then A end



- begin if not C then A end

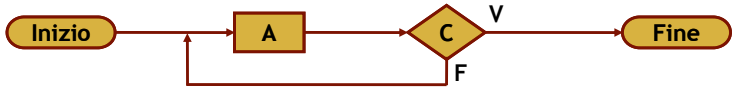
Linguaggio Lineare Iterazione (while…do)

- begin while C do A end

- begin while not C do A end

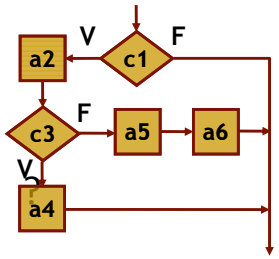
Linguaggio Lineare
Iterazione (repeat…until)

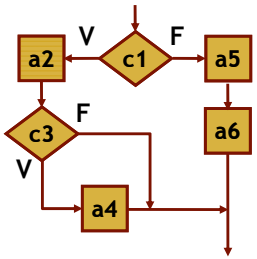
- begin repeat A until C end



- begin repeat A until not C end

Linguaggio Lineare
Ambiguità

- if c1 then a2; if c3 then a4 else a5; a6
- 

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 le descrizione di un sottoprocesso che segue le parola chiave
    - then
    - else
    - while
  - quando non è un'azione basica

Risoluzione delle Ambiguità
Convenzioni aggiuntive

- In alternativa:
  aggiungere i seguenti terminatori 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

Esempio
Algoritmo Euclideo per il MCD

- Leggi la coppia di numeri
- Fintantoché i numeri sono diversi ripeti
  - Se il primo numero è maggiore del secondo allora
    - Calcola la differenza tra primo e secondo
  - altrimenti
    - Calcola la differenza tra secondo e primo
  - Rimpiazza i due numeri col sottraendo e con la differenza, rispettivamente
- Il risultato è il valore dei due numeri della coppia (uguali)

Esempio
Algoritmo Euclideo per il MCD

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