Programmazione Strutturata

Problema: Spaghetti Code

- è un termine dispregiativo per quei programmi per computer che abbiano una struttura di controllo del flusso complessa e/o incomprensibile
- Ad esempio, la presenza di salti crea flussi di controllo non predicibili e dipendenti dai numeri attribuiti ai passi del programma

Spaghetti code: stampa il quadrato dei primi 10 interi

```
1. INIZIO
2. Poni i = 0
3. incrementa i di uno
4. Se i è diverso da 10 vai al punto 9
5. Se i = 10 vai al punto 7
6. Vai al punto 3
7. stampa "Programma terminato."
8. Esci dal programma
9. stampa i * i
10.Vai al punto 6
```

Spaghetti code: controesempio strutturato

- L'introduzione di un blocco (`bloccoA`) avente un solo punto di ingresso/uscita, unito ad istruzioni di iterazione di piu' facile comprensione.

```
1.    INIZIO
2.    Poni i = 1
3.    Esegui bloccoA finchè i < 10
4.    Inizio bloccoA
5.    Incrementa i di uno
6.    Stampa i*i
7.    Fine bloccoA
8.    stampa "Programma terminato."
```

Programmazione strutturata

- La programmazione strutturata nasce come proposta per regolamentare e standardizzare le metodologie di programmazione (Dijkstra, 1965)

- **Obiettivo:**
  - rendere più facile la lettura dei programmi (e quindi la loro modifica e manutenzione)
- **Idea di base:**
  - La parte esecutiva di un programma viene vista come un comando (complesso o *strutturato*) ottenuto componendo **istruzioni elementari**, mediante alcune regole di composizione (**blocco, selezione, ciclo**)

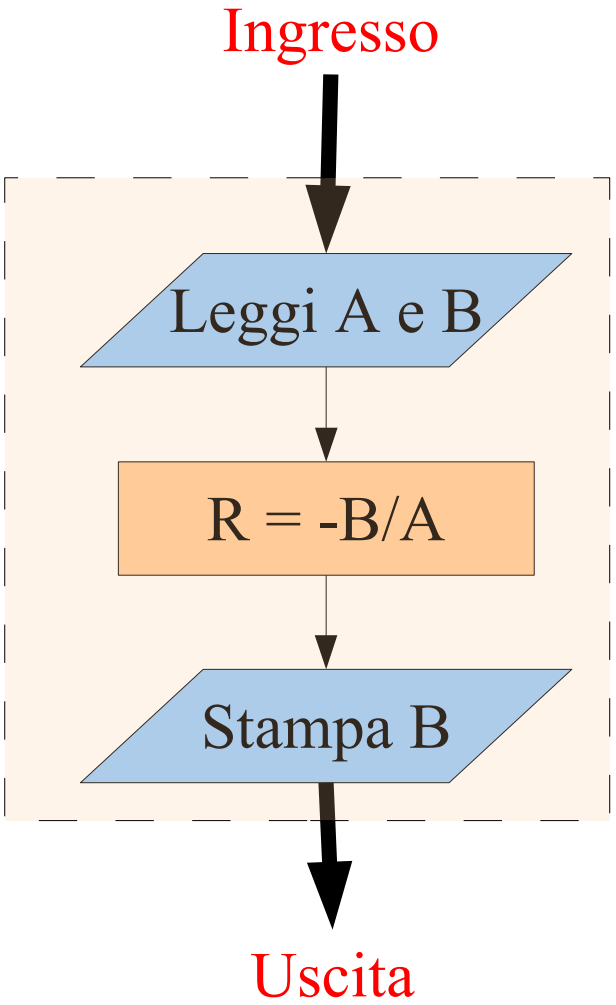
Programmazione strutturata

Concetti chiave:

- **BLOCCO:** sequenza o composizione di istruzioni, avente un solo ingresso e una sola uscita
- **SELEZIONE:** struttura condizionale per la scelta di esecuzione tra blocchi
- **CICLO:** struttura di ripetizione (o iterazione) di un blocco

NB: Abolizione di **salti incondizionati** (***goto***) nel flusso di controllo! (i "*vai al punto X*" dell'esempio precedente)

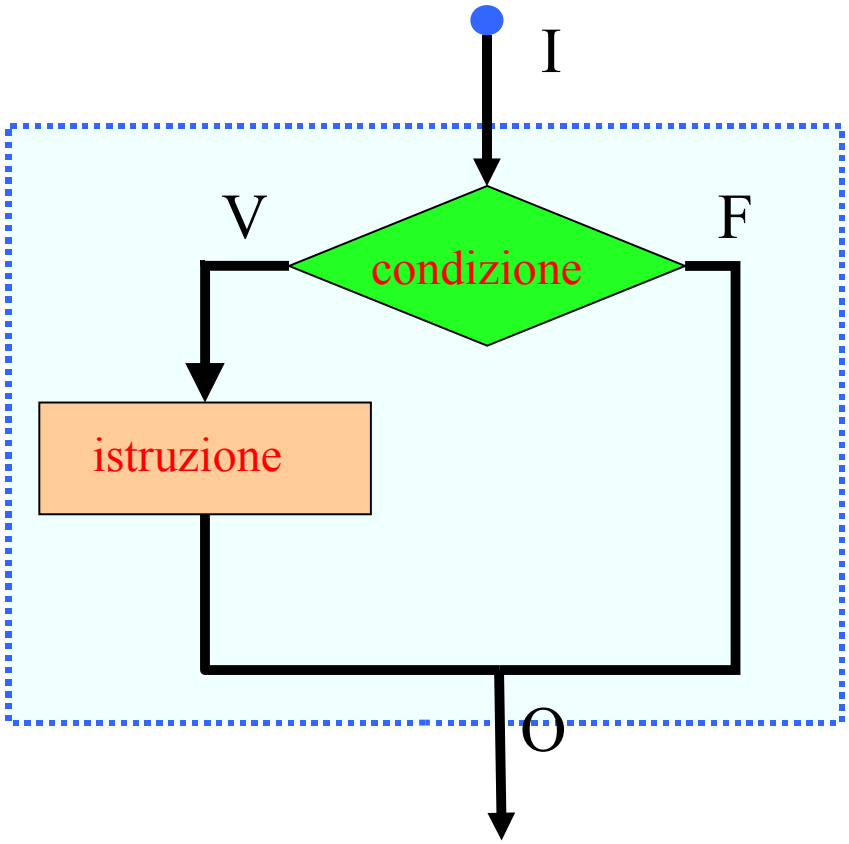
Struttura Blocco (o sequenza)



- Un solo ingresso e una sola uscita
- Durante il blocco non ci sono salti che escono fuori dal blocco, oppure salti che da blocchi esterni entrano nel blocco
- Le istruzioni verranno eseguite sequenzialmente.
- Il concetto di blocco permette di considerare *la sequenza di più istruzioni come un'unica istruzione* (composta)

Struttura selezione *(a una via)*

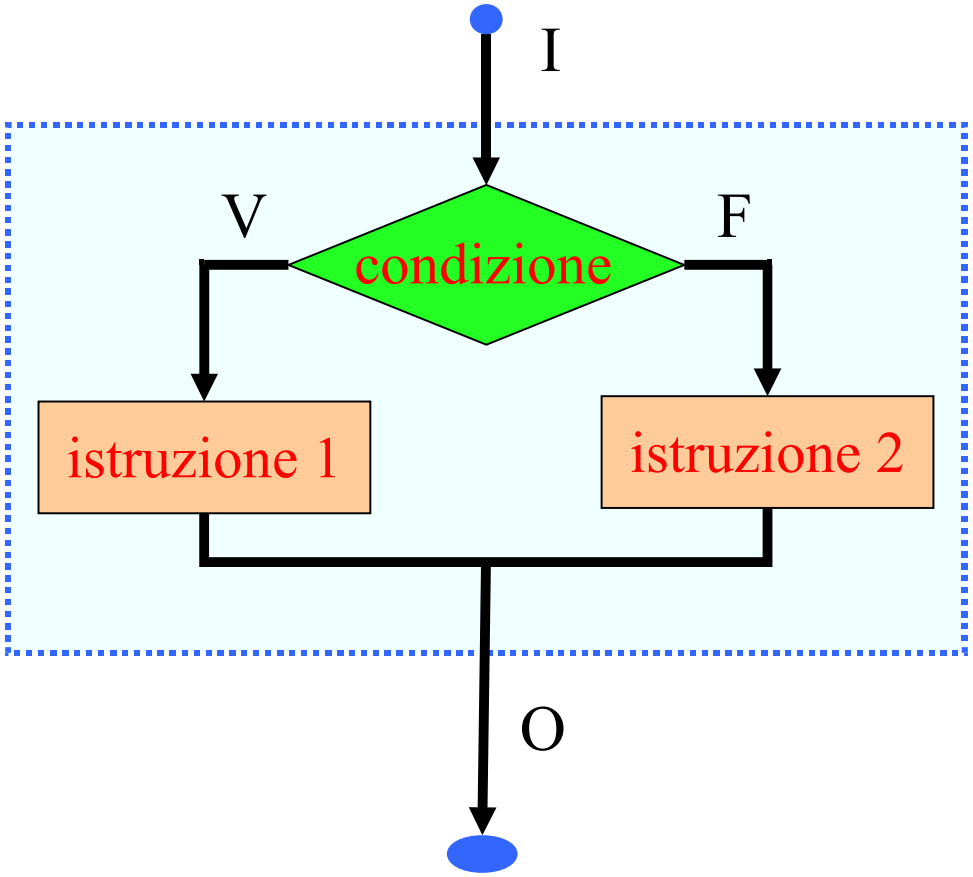
Diagramma a blocchi



Se la **condizione** è vera allora viene eseguita **istruzione** altrimenti no.
NB: *per quanto detto, l'istruzione puo' essere un blocco di piu' operazioni*

Struttura selezione *(a due vie)*

Diagramma a blocchi

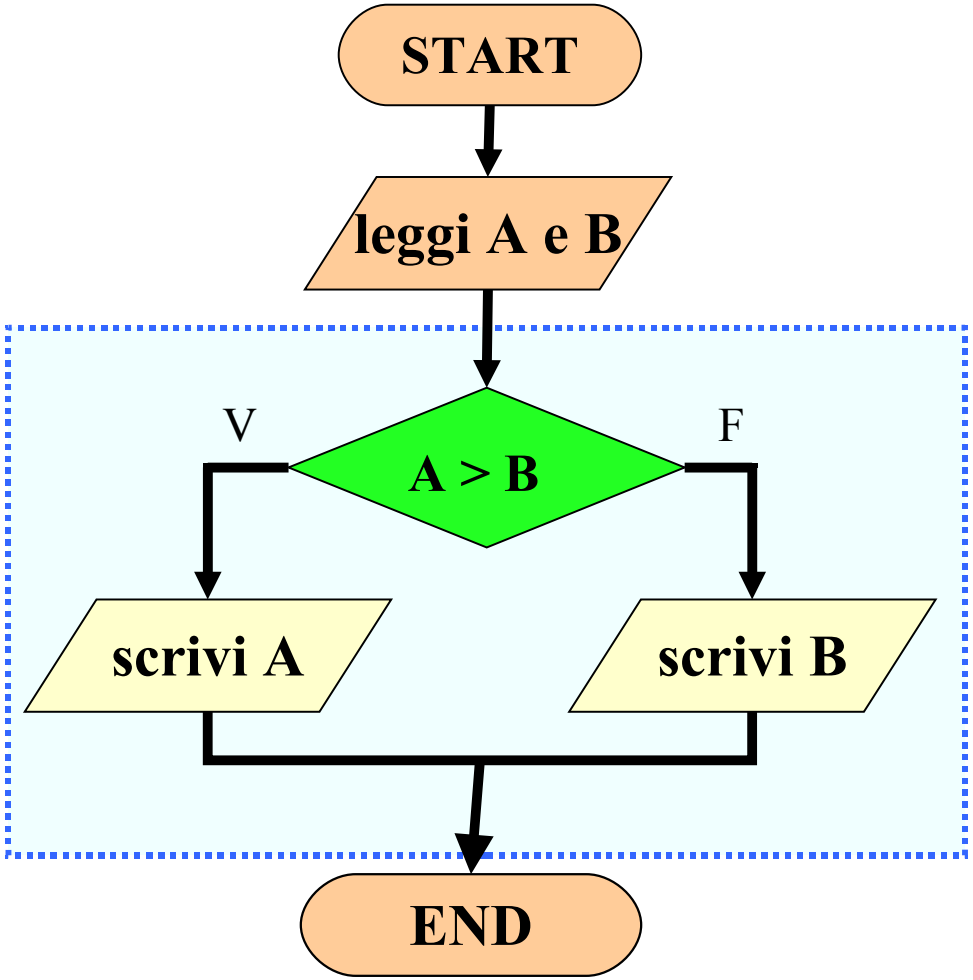


Se la **condizione** è vera allora viene eseguita **istruzione 1** altrimenti viene eseguita **istruzione 2**

Strutture di selezione *(osservazioni ed esempi)*

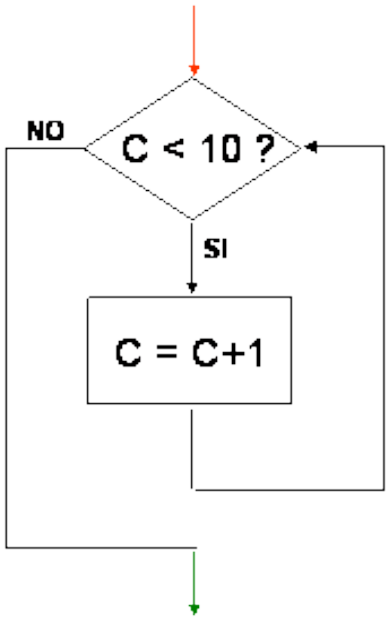
Es: determina e stampa il maggiore tra due numeri letti da input

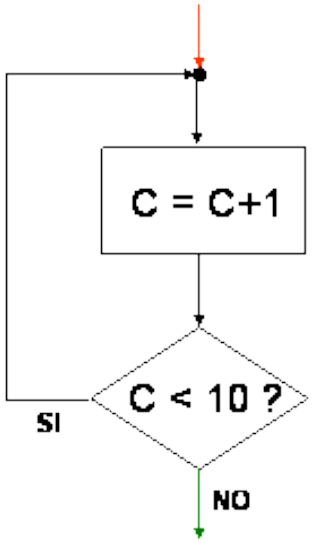
Domanda: funziona se A e B sono uguali ?



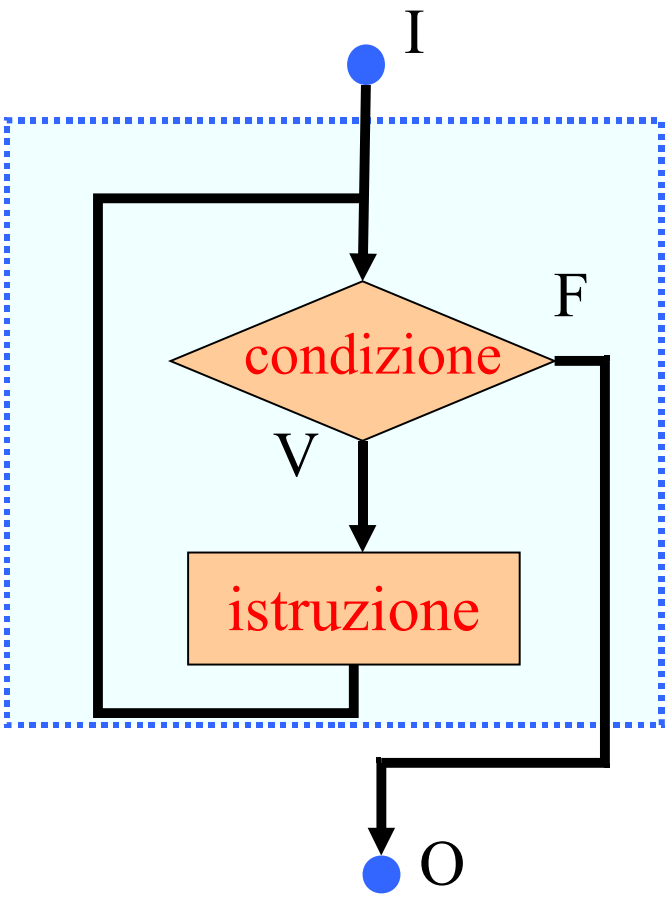
Strutture di Iterazione

- Due possibilità: controllo della condizione di iterazione all'inizio (while) o alla fine del blocco (do...while)





Struttura di iterazione *(while)*



- <istruzione> viene ripetuta

  per tutto il tempo in cui la condizione rimane ***vera***

- Se la condizione è già inizialmente falsa, l'iterazione non viene eseguita *neppure una volta*

- In generale, *non è noto* a priori ***quante volte*** l'istruzione sarà ripetuta

Struttura di iterazione *(while)*

- Prima o poi, *direttamente o indirettamente*, l'istruzione deve *modificare la condizione:* altrimenti → ***CICLO INFINITO***

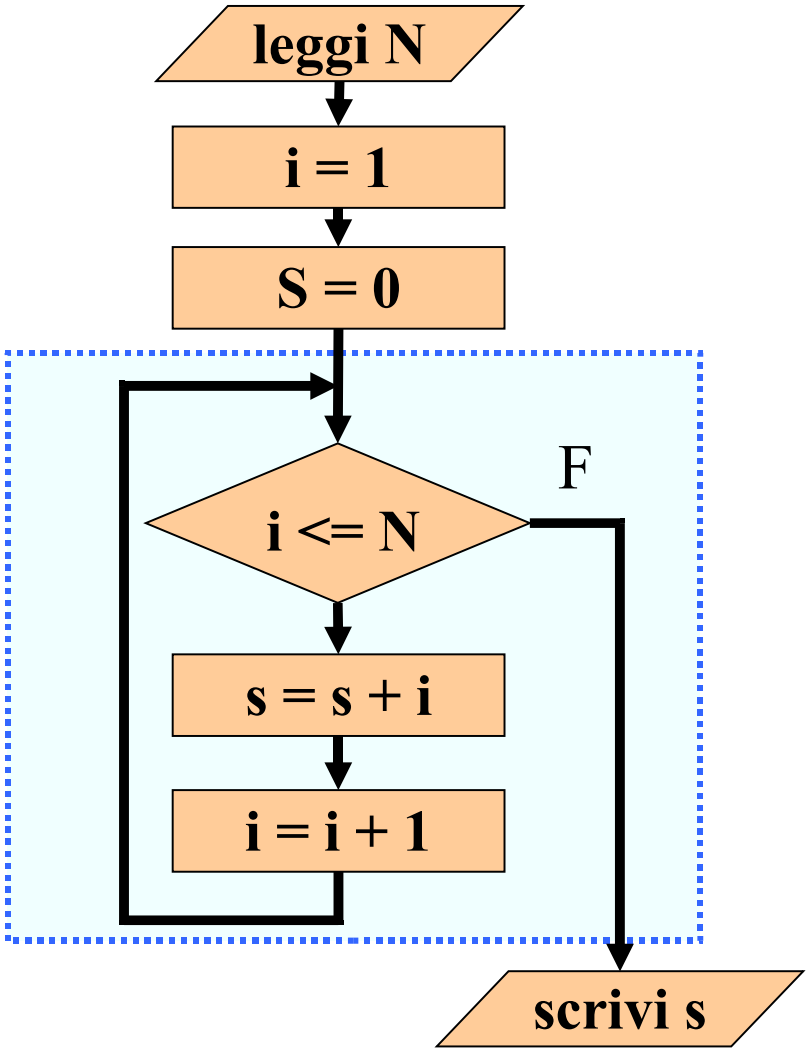
- Per questo motivo, quasi sempre *<istruzione>* è in realtà un blocco, che contiene una istruzione in cui si modifica qualche variabile che compare nella condizione

Struttura di iterazione *(esempi)*

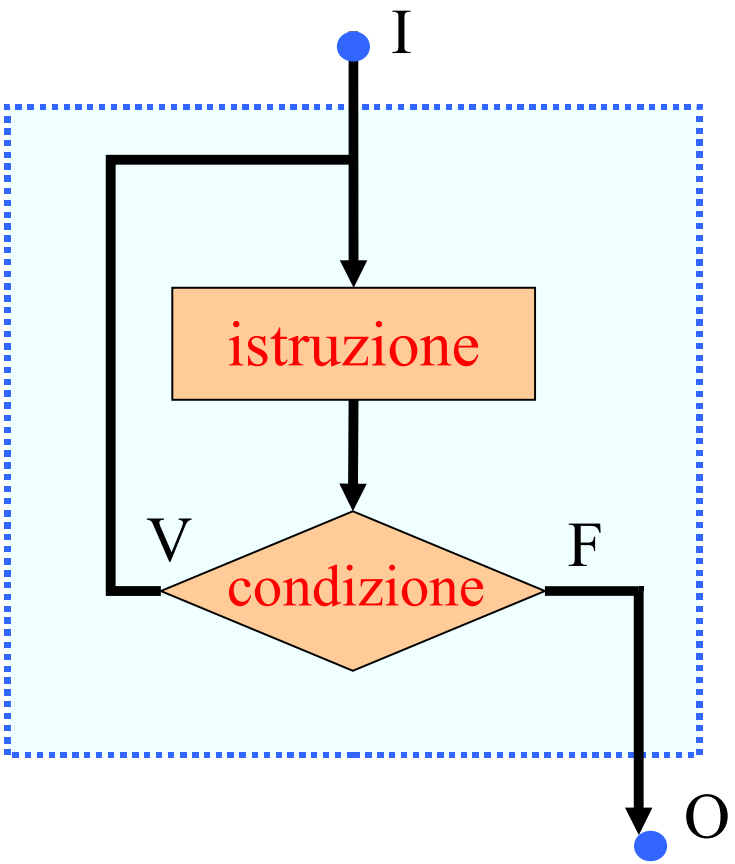
Es: Letto un numero intero N da input, sommare i primi N numeri positivi e scrivere il risultato

Domanda:

- che blocchi ci sono ?
- Che cicli ?
- Che selezioni ?



Struttura di iterazione *(do-while)*



- È una variante della precedente: la condizione viene verificata ***dopo*** aver eseguito <istruzione>
- Se la condizione è falsa, l'istruzione **viene comunque eseguita *almeno una volta***

Struttura di iterazione *(do-while)*

- Analogamente al *while*, per evitare il ciclo infinito, <istruzione> deve modificare prima o poi qualche variabile che compare nella condizione
- Si noti che, come nel caso del *while*, si esce dal ciclo quando la condizione è falsa
- **È adatta** a quei casi in cui, per valutare condizione, è necessario aver già eseguito <istruzione>

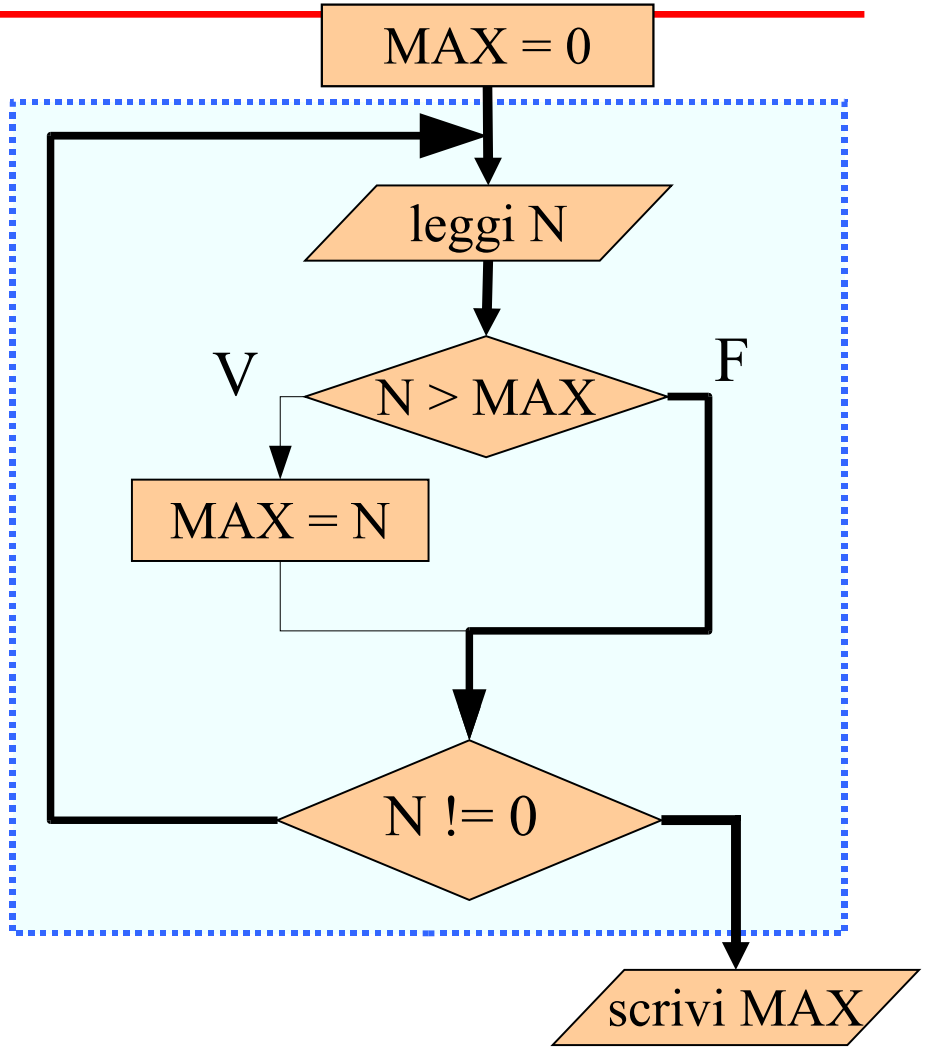
  (esempio tipico: controllo di valori di input)
- **Non è adatta** a quei casi in cui il corpo del ciclo può non dover essere *mai eseguito*

Struttura di iterazione *(esempi)*

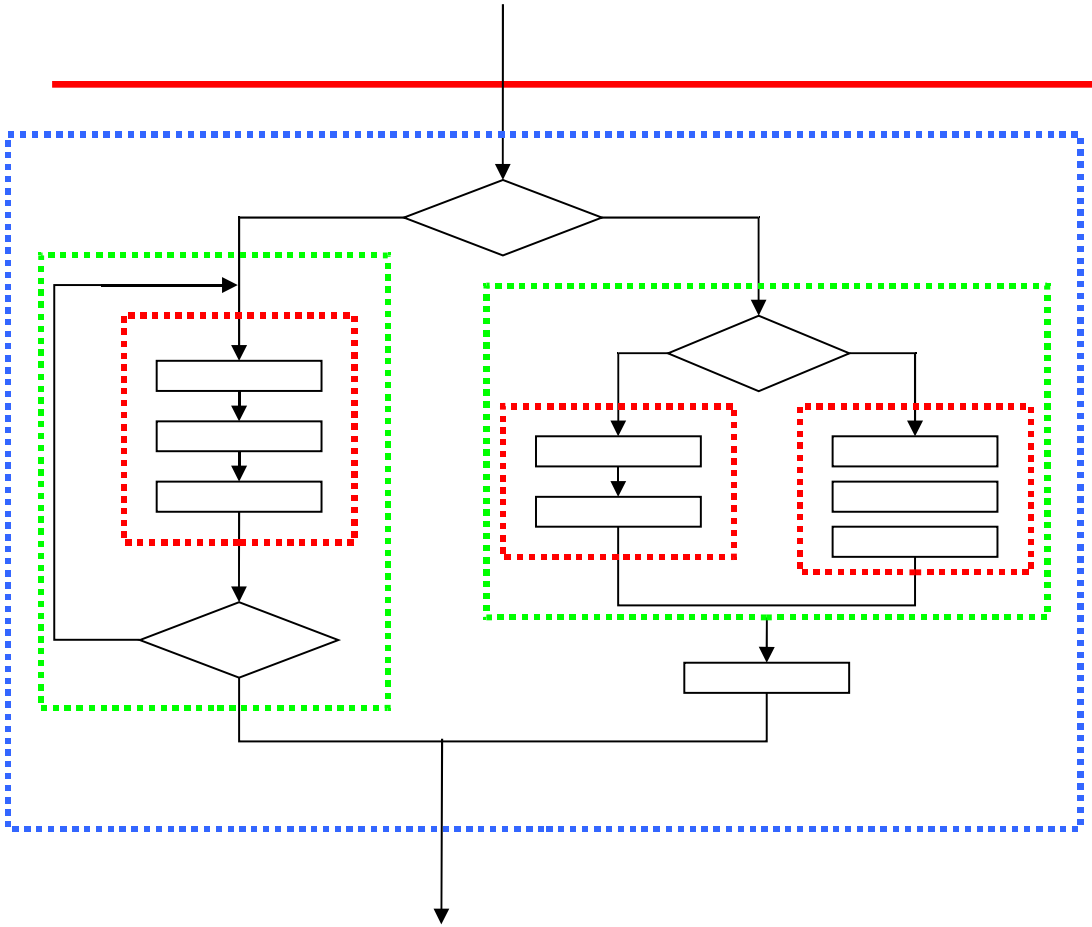
Es:Scrivere il valore massimo di una sequenza di interi positivi (letti da input) chiusa da uno zero

Domande:

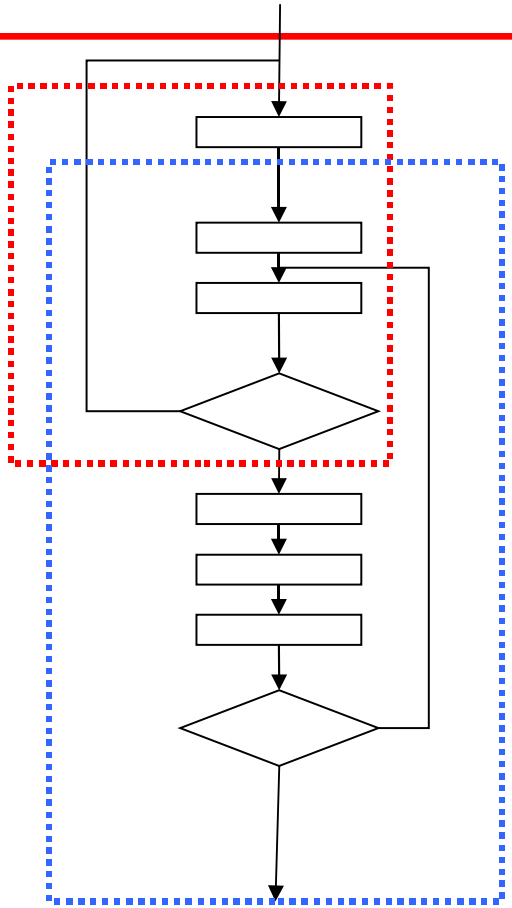
- qual e' la istruzione/blocco che viene ripetuta dalla iterazione ?
- Ci sono selezioni ?



Programmazione strutturata



Corretto



Sbagliato:

Non ci sono cicli e/o selezioni su blocchi

Un dubbio...

- Ma queste restrizioni, cioè l'utilizzo di solamente selezione e iterazione su blocchi, senza l'uso dei salti, dà la stessa "potenza di calcolo" (espressività) oppure si perde qualcosa?

Teorema di Bohm- Jacopini (1966)

(in versione semplificata)

Le strutture di sequenza blocco, selezione e iterazione sono sufficienti ad esprimere un qualsiasi algoritmo

Ossia: L'uso di queste sole strutture non limita il potere espressivo...siamo salvi!

Alcune considerazioni finali

- Tutti i linguaggi imperativi implementano una qualche forma delle strutture presentate
- Naturalmente la forma sintattica può a volte variare leggermente, ma il funzionamento rimane identico
  - uso di diversi marcatori per l'inizio e la fine di un blocco (*begin* e *end* in Pascal)
  - uso della parola chiave *then* nell'istruzione if (in Pascal)
- Inoltre i vari linguaggi introducono altre strutture di controllo, non indispensabili, ma atte a semplificare il lavoro di scrittura del codice
  - istruzione di scelta multipla (*switch-case* in C)
  - istruzione iterativa controllata da contatore (*for*)

Programmazione strutturata:Vantaggi

- Leggibilità (vedi spaghetti code...)
- Supporto a metodologie di progetto top-down: Soluzione di problemi complessi attraverso la scomposizione in sotto-problemi, a loro volta scomponibili in sotto-problemi, etc.
- Supporto a metodologia bottom-up: La soluzione di problemi avviene aggregando componenti già disponibili mediante concatenazione, selezione e ripetizione
- Maggior facilità di verifica e manutenzione