Strutture di controllo

(9')

Labirinto classico

In **JavaScript** (=>Wk)

```
moveForward();
moveForward();
turnRight();
moveForward();
```

Sequenza

Selezione multipla

```
while (notFinished()) {
  if (isPathForward()) {
    moveForward();
  } else {
    turnLeft();
  }
}
```

Selezione semplice

```
while (notFinished()) {
  moveForward();
  if (isPathLeft()) {
    turnLeft();
  }
}
```

Ripetizione (o iterazione o ciclo)

for che si ripete per 4 volte con corpo moveForward()

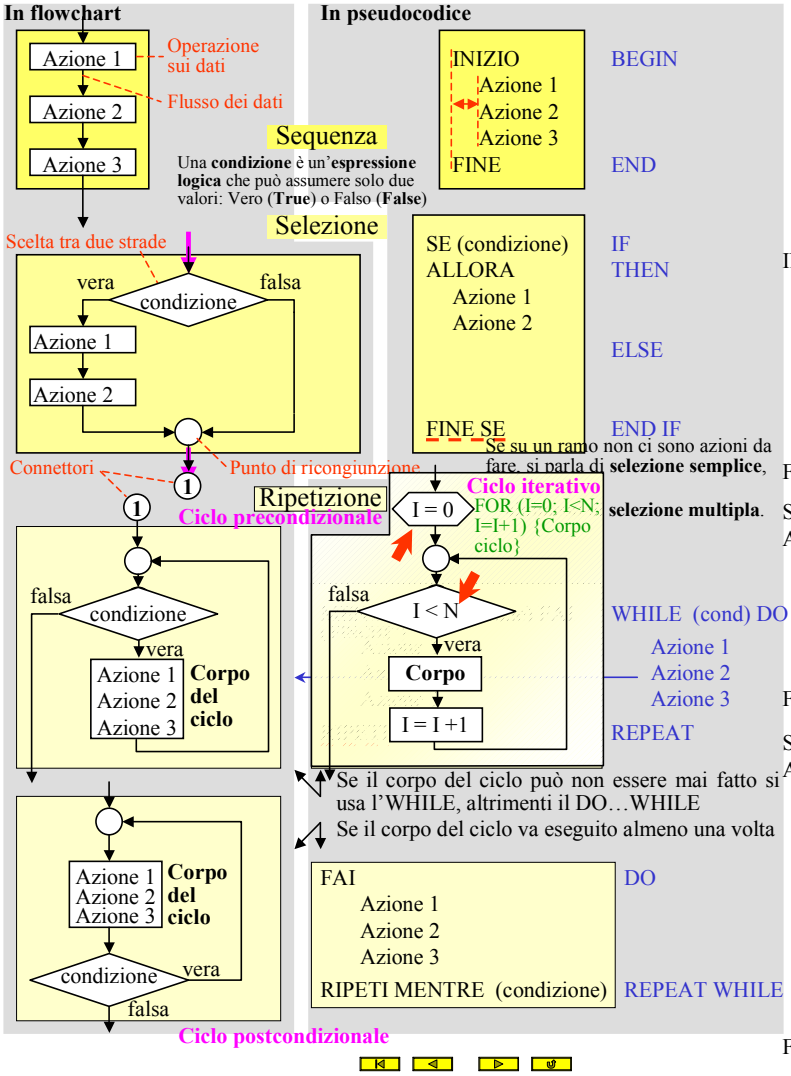
```
for (var count = 0; count < 4; count++) {
  moveForward();
}
```

Se non è un vettore di N elementi allora… il ciclo standard è un **while** (**mentre** è vera la condizione tra parentesi tonde fai il corpo del ciclo)

```
while (notFinished()) {
  moveForward();
}
```

Vediamo come si scrivono, in flow-chart e nello pseudocodice adottato dal libro Lorenzi-Moriggia, le 3 strutture fondamentali che il **teorema di Böhm-Jacopini** assicura essere sufficienti per scrivere algoritmi senza usare mai istruzioni di salto(il famigerato GOTO… capace di rendere il programma un 'piatto di spaghetti' difficilissimo da capire e da correggere)

In flowchart | **In pseudocodice**



I simboli di inizio e di fine dell'algoritmo del flow-chart (che corrispondono al BEGIN…END dello pseudocodice) sono rettangoli stondati.

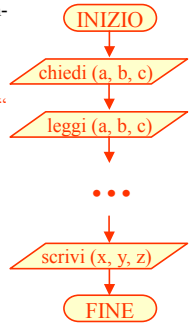
Vista la particolare importanza delle operazioni di acquisizione dei dati d'ingresso e di comunicazione dei dati finali o risultati, per tali operazioni il flow-chart usa un simbolo apposito: il parallelogramma.

L'azione "chiedi (a)" *è di fatto una operazione di uscita tipo "scrivi (m)"*, solo che m non è un risultato ma un messaggio per chi è davanti al computer (tipo "Dammi il valore di a: "). L'azione "leggi (a)" consisterà nel leggere quanto battuto dall'utente *e nell'assegnarlo alla variabile a*.

In psuedocodifica si useranno le stesse parole e la sola differenza è che manca la grafica

chiedi (a) = scrivi (m)
dove m è un messaggio "Dammi il valore di a: "

```
INIZIO
  chiedi (a,b,c)
  leggi (a,b,c)
  • • •
  scrivi (x,y,z)
FINE
```




```
INIZIO
  Solleva ricevitore
  Ascolta segnale
  SE (segnale linea presente)
  ALLORA
    telefona
  FINE SE
  Posa ricevitore
FINE

SE (segnale linea presente)
ALLORA
  Componi numero
  Ascolta segnale
  SE (non occupato)
  ALLORA
    Aspetta o dialoga
FINE SE

SE (non occupato)
ALLORA
  MENTRE (non c'è segn. l.)
  FAI
    SE (utente risponde)
    ALLORA
      Dialoga
    ALTRIMENTI
      Ascolta segnale
    FINE SE
  RIPETI
FINE SE
```

```
INIZIO
  Solleva ricevitore
  Ascolta segnale
  SE (segnale linea presente)
  ALLORA                    Telefona
    Componi numero
    Ascolta segnale
    SE (non occupato)
    ALLORA                  Aspetta o dialoga
      MENTRE (non c'è segnale di libero)
      FAI
        SE (utente risponde)
        ALLORA
          Dialoga
        ALTRIMENTI
          Ascolta segnale
        FINE SE
      RIPETI
    FINE SE
  FINE SE
  Posa ricevitore
FINE
```

Programmare in pseudocodice diventa facile e naturale se si procede per raffinazioni successive (**programmazione TOP-DOWN**)

Per verificare se è un **diagramma di flusso strutturato** basta *ridurlo* sostituendo rettangoli alle selezioni e ai cicli.

Se *alla fine si riduce ad un solo rettangolo*, era strutturato contenendo evidentemente solo rettangoli, selezioni e cicli