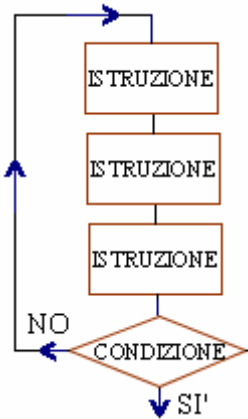


### 3 - DIAGRAMMI DI FLUSSO; PROGRAMMAZIONE STRUTTURATA



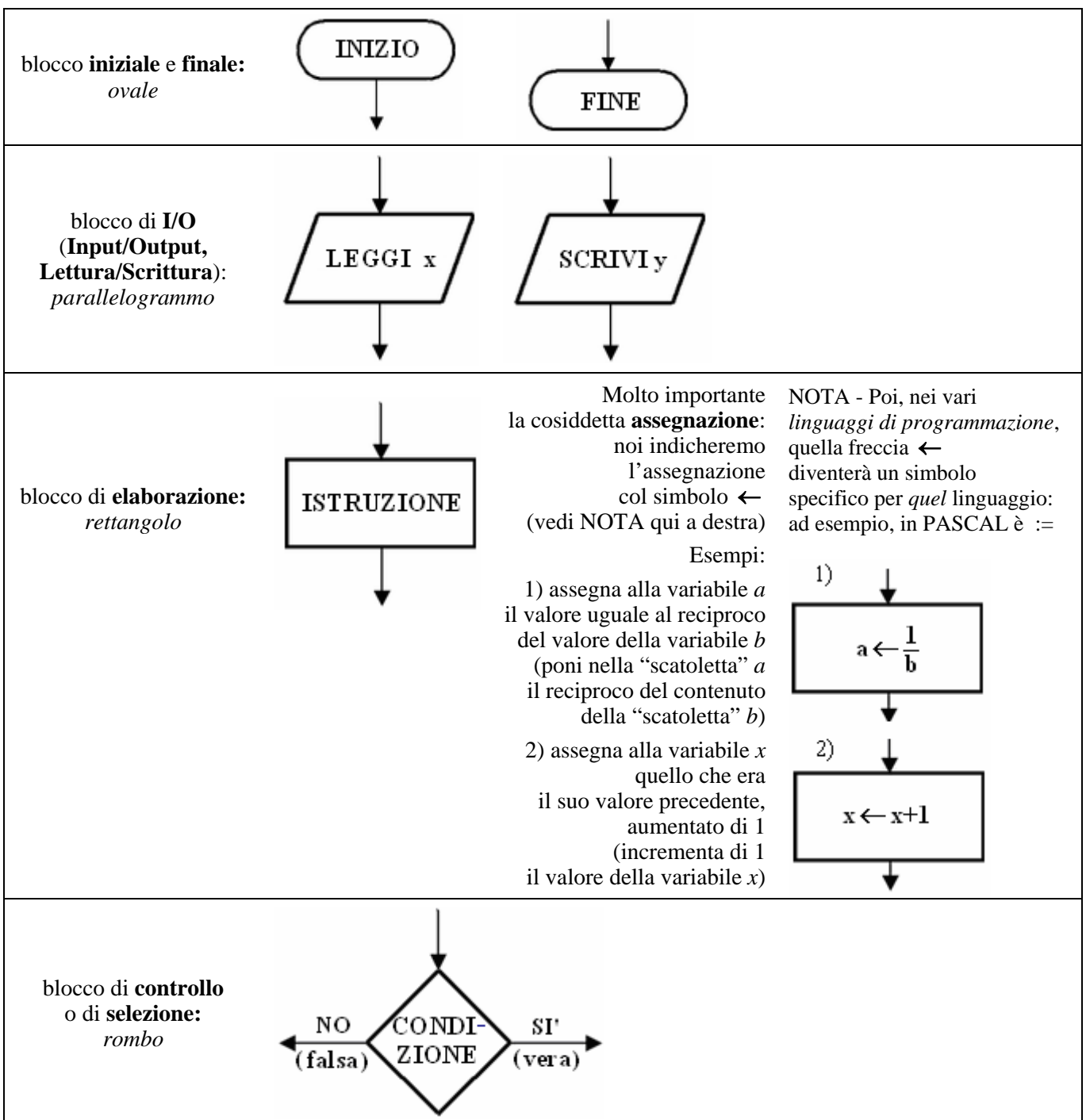
Lo schema qui a sinistra, che abbiamo utilizzato per descrivere il “funzionamento” di una REPEAT ... UNTIL ... del linguaggio Pascal, è un esempio di **DIAGRAMMA DI FLUSSO**.

Per “diagramma di flusso” (o “diagramma a blocchi”; in lingua Inglese: “flow chart”) si intende dunque **la raffigurazione di un algoritmo attraverso blocchi, collegati da linee orientate**.

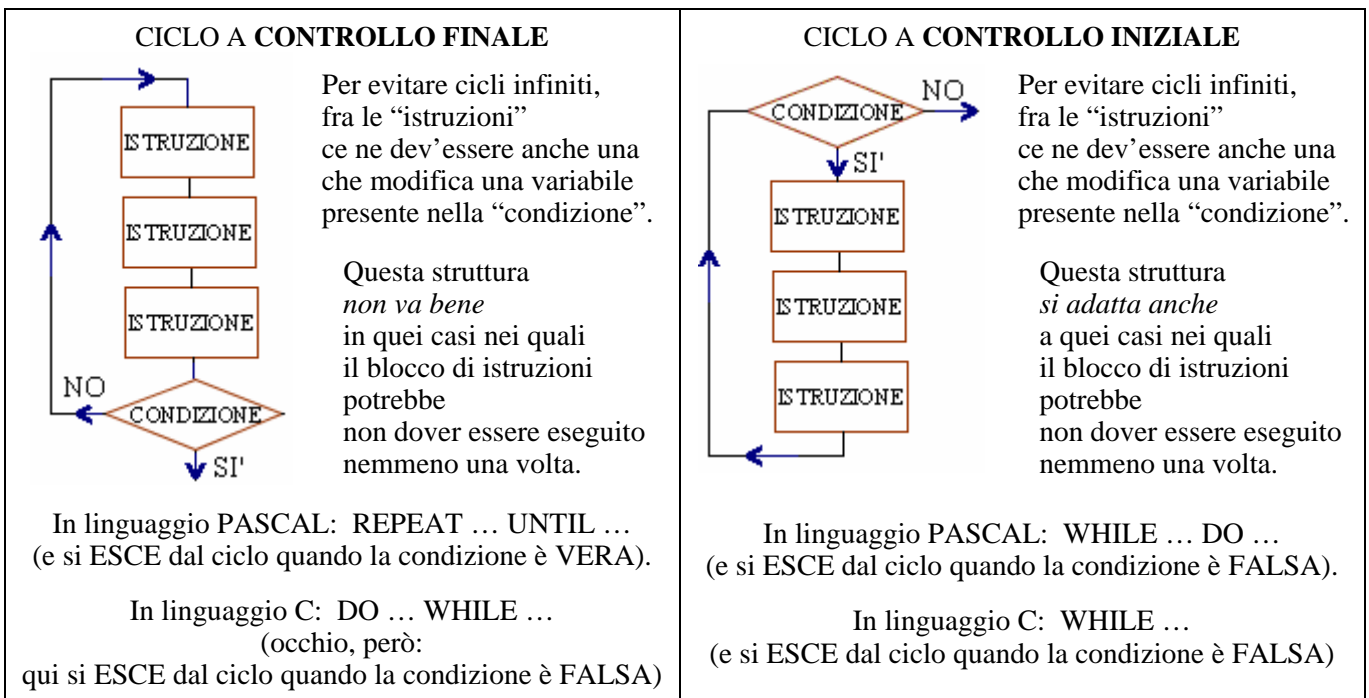
Ciascun blocco contiene una istruzione, oppure una condizione.

Per la forma dei “blocchi” si sono affermate certe **convenzioni**.

Precisamente:



E' opportuno organizzare il diagramma di flusso per la risoluzione di un dato problema in modo che, se c'è una "freccia che ritorna indietro", ossia: se c'è una *ripetizione*, una **ITERAZIONE** di istruzioni, questa iterazione sia governata da una delle due strutture seguenti:

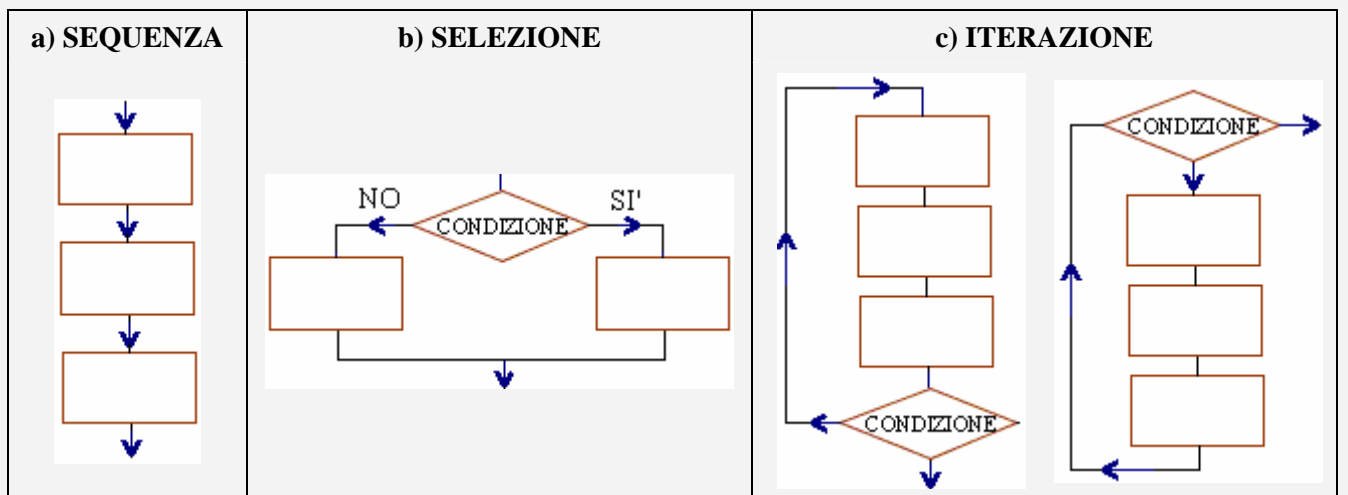


Conviene invece evitare l'inserimento nel diagramma di flusso di frecce che facciano "salti incondizionati" da un punto all'altro (istruzioni del tipo "go to").

Tale uso disordinato di "salti" (chiamato "**spaghetti programming**") è infatti considerato *controproducente*,

- ❑ sia ai fini del controllo della correttezza dell'algorithm,
- ❑ sia per una eventuale successiva traduzione del diagramma in un programma per computer.

D'altra parte un famoso enunciato, il **teorema di Bohm-Jacopini** (1966), afferma che **qualunque algoritmo si può realizzare con le sole tre strutture di**



(nei rettangoli possiamo trovare singole istruzioni oppure altre strutture degli stessi 3 tipi)

E una "buona" programmazione, che eviti i "salti incondizionati" e sia invece basata sul teorema di Bohm-Jacopini ovvero sulle tre strutture di *sequenza, selezione e iterazione* (ciascuna con una sola uscita, con possibilità di "nidificarle" una dentro l'altra ma NON di disporle in modo che si "accavallino"), è chiamata "**programmazione STRUTTURATA**".

La programmazione strutturata rende più ordinata la stesura di un programma, e ne rende più semplici la lettura e il controllo di correttezza, nonché l'eventuale completamento o utilizzo col ruolo di "sottoprogramma" (metodi di lavoro *top-down* o *bottom-up*).