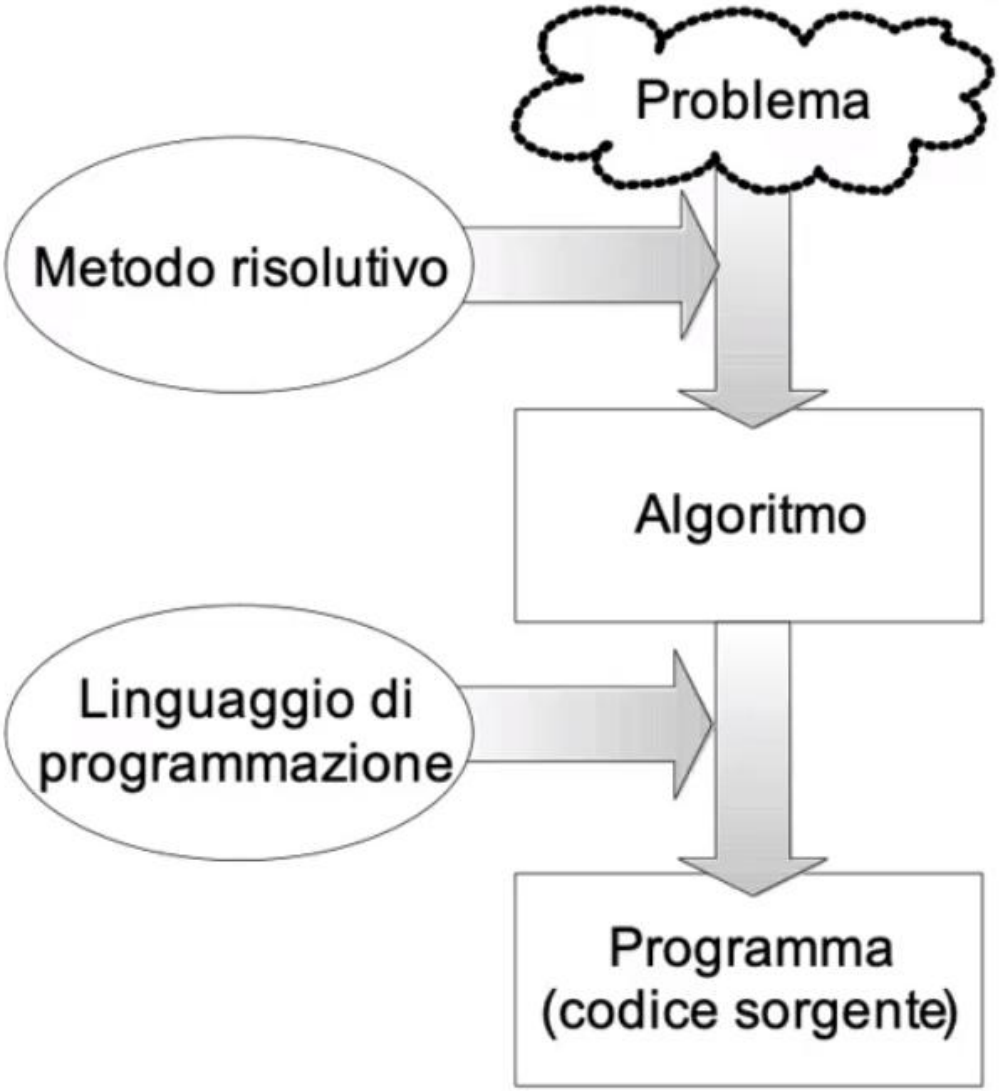
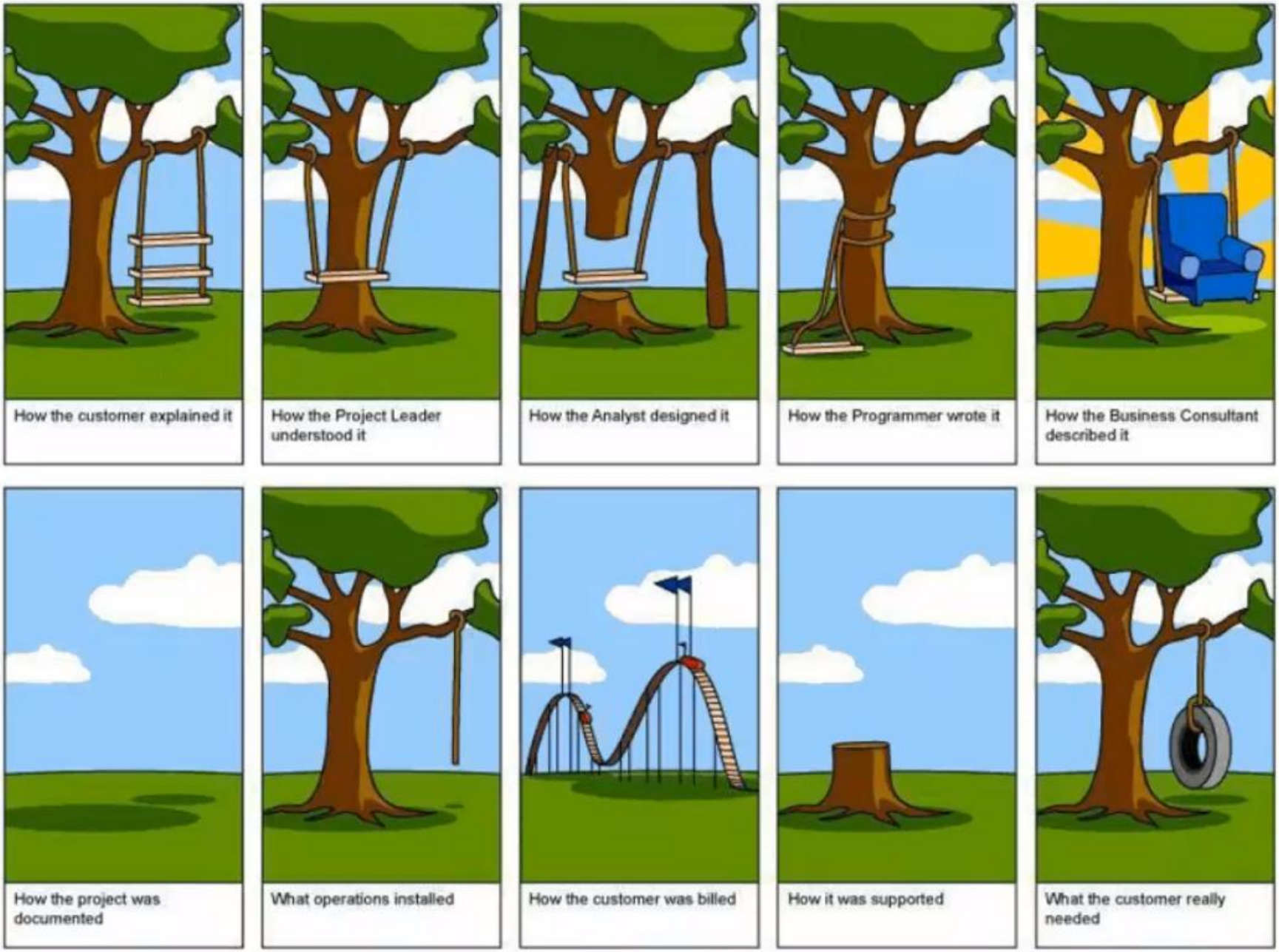
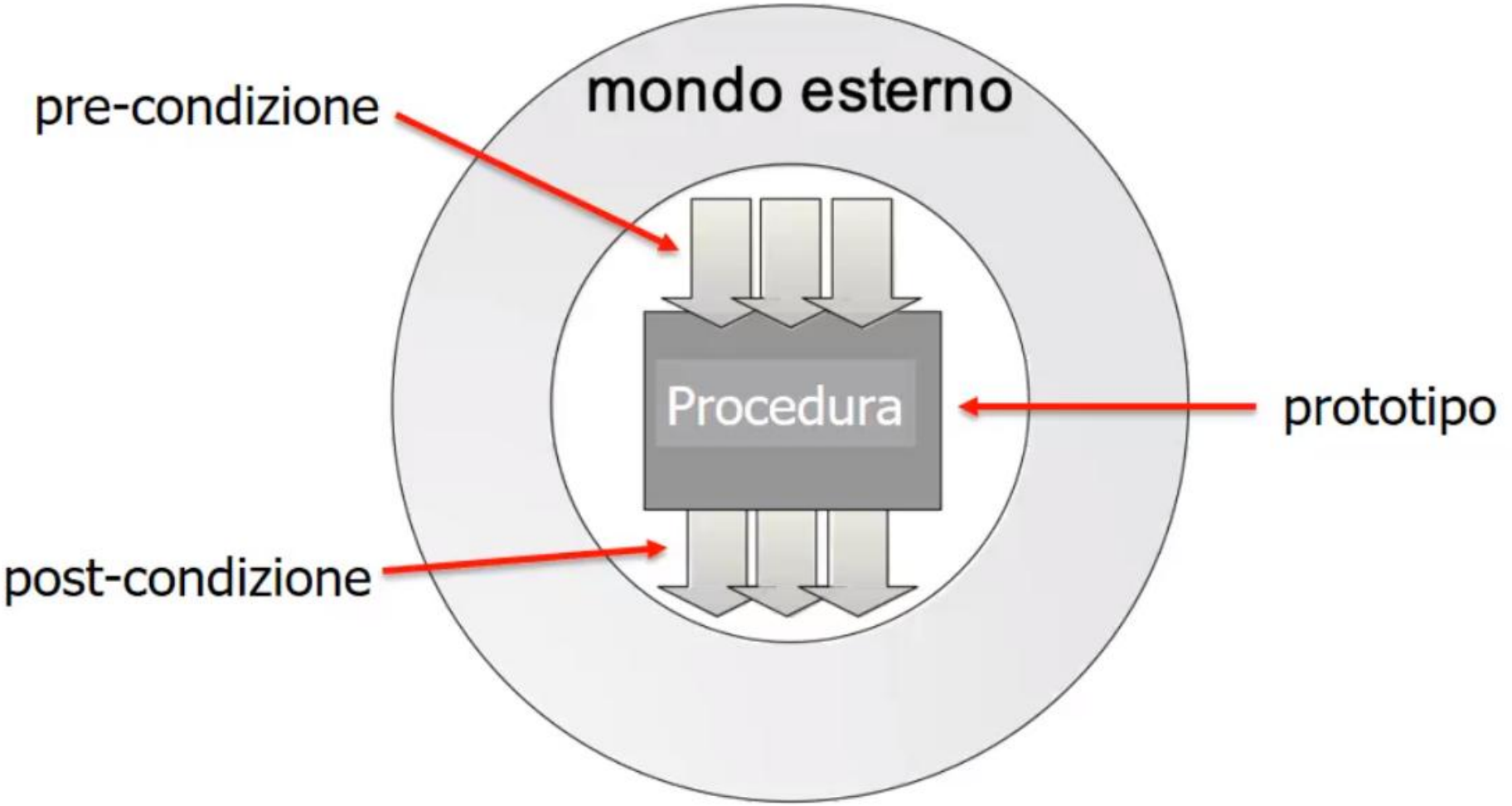
Passi da seguire



https://wiki.lokar.fmf.uni-lj.si/diriwiki/index.php/Kak%C5%A1no_programsko_opremo_stranka_resni%C4%8Dno_potrebuje



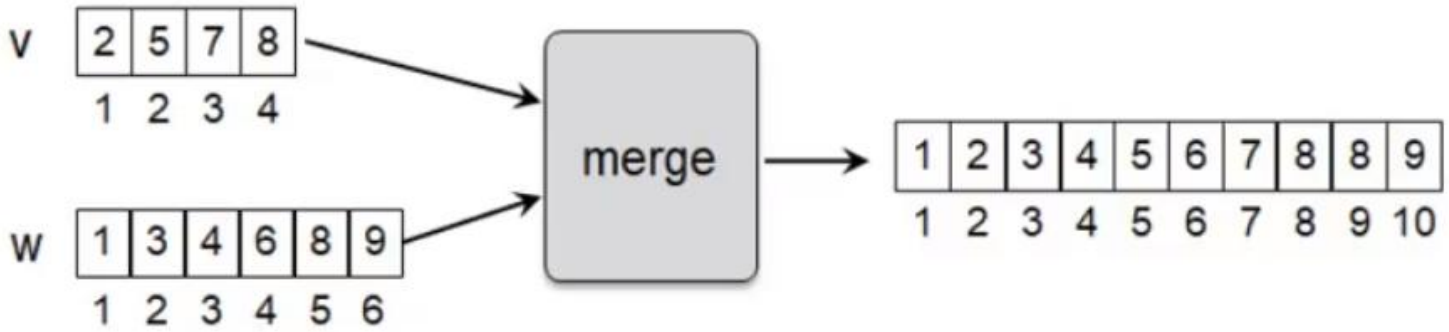
Il «contratto»



Requisiti chiari...

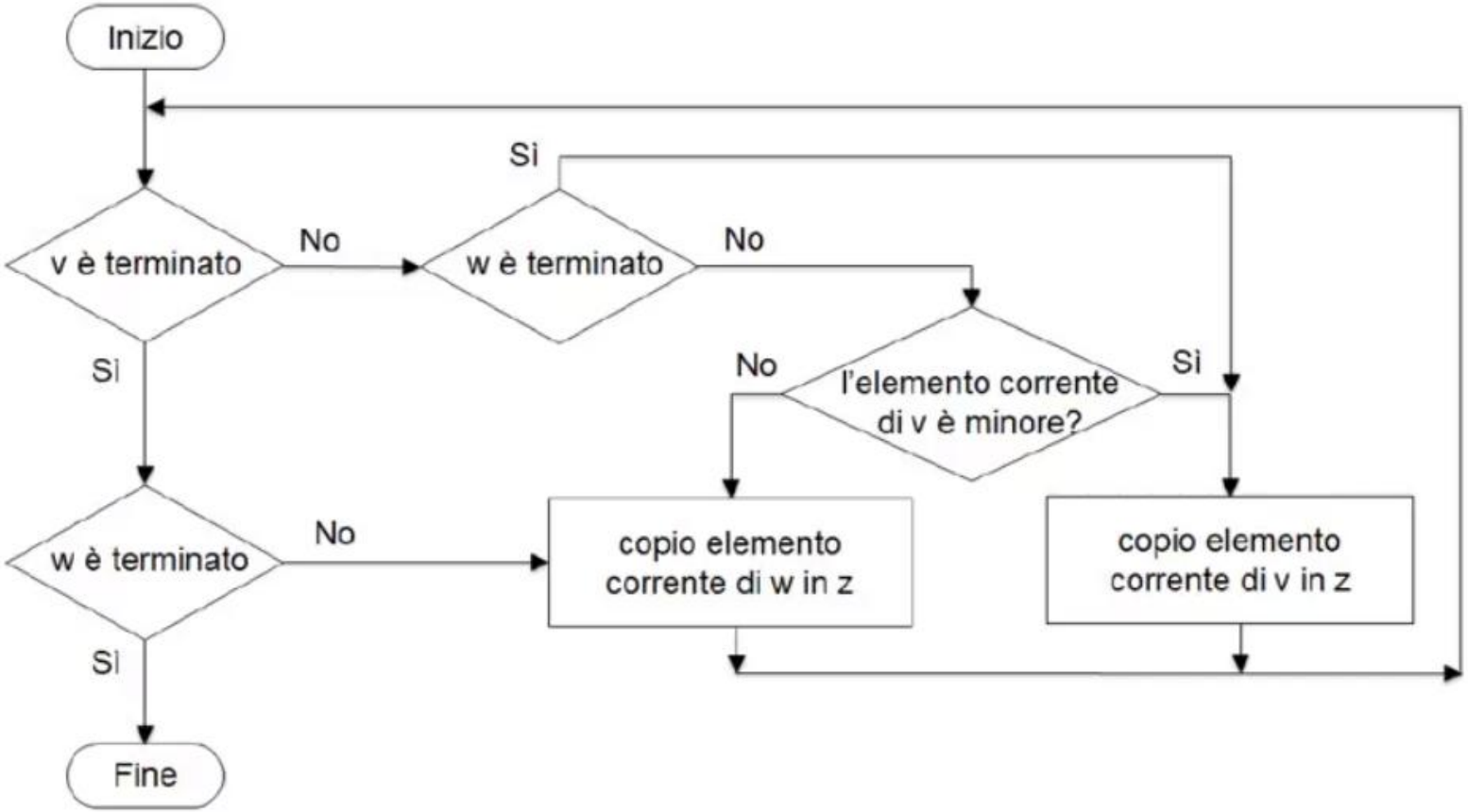
Esercizio

Descrivere requisiti (obiettivo e contratto) e metodo dell'algoritmo che riceve in ingresso due array ordinati in modo crescente (v e w) e restituisce in uscita un nuovo array con all'interno l'insieme ordinato degli elementi dei due array originari .

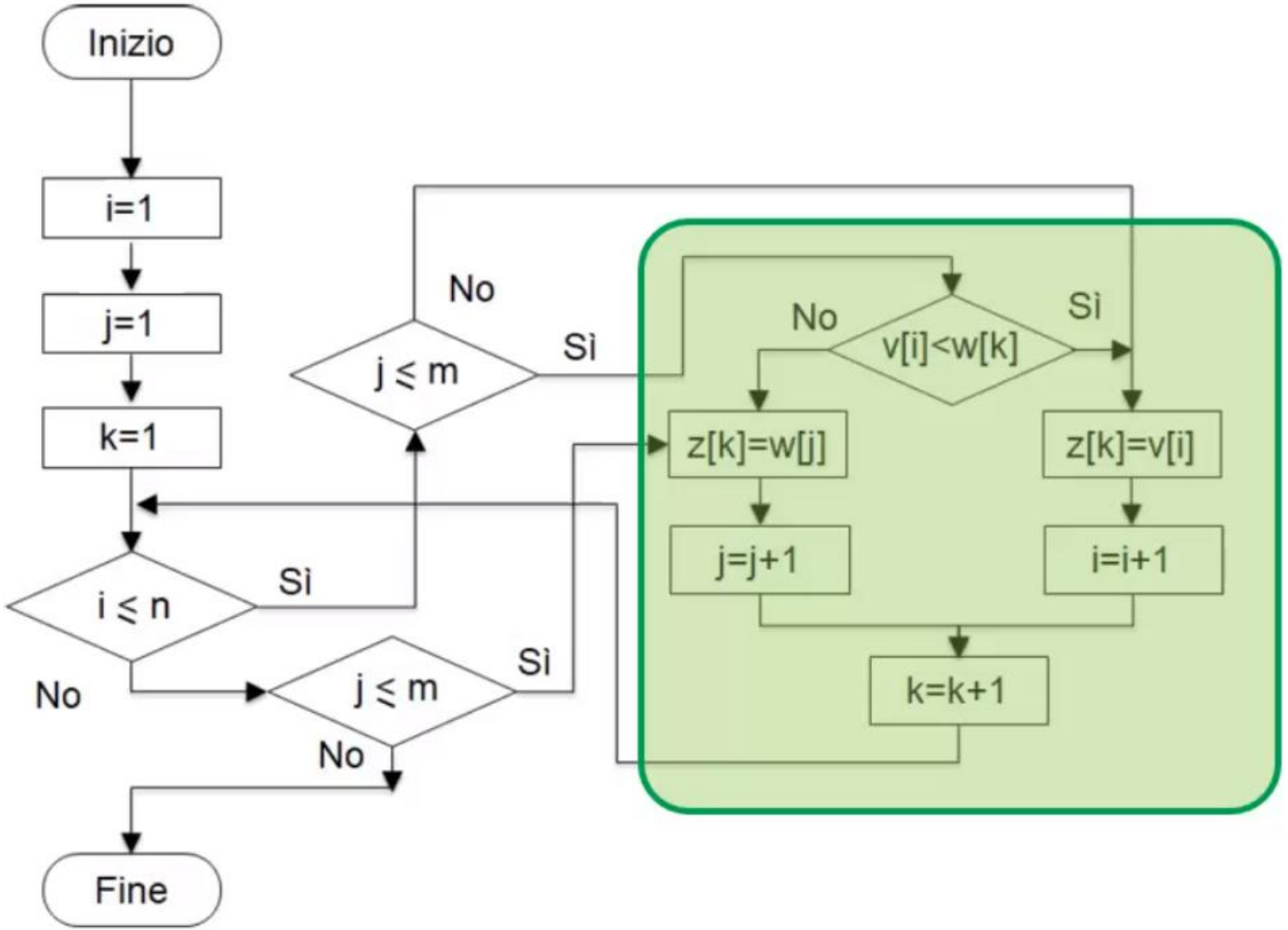


Esercizio

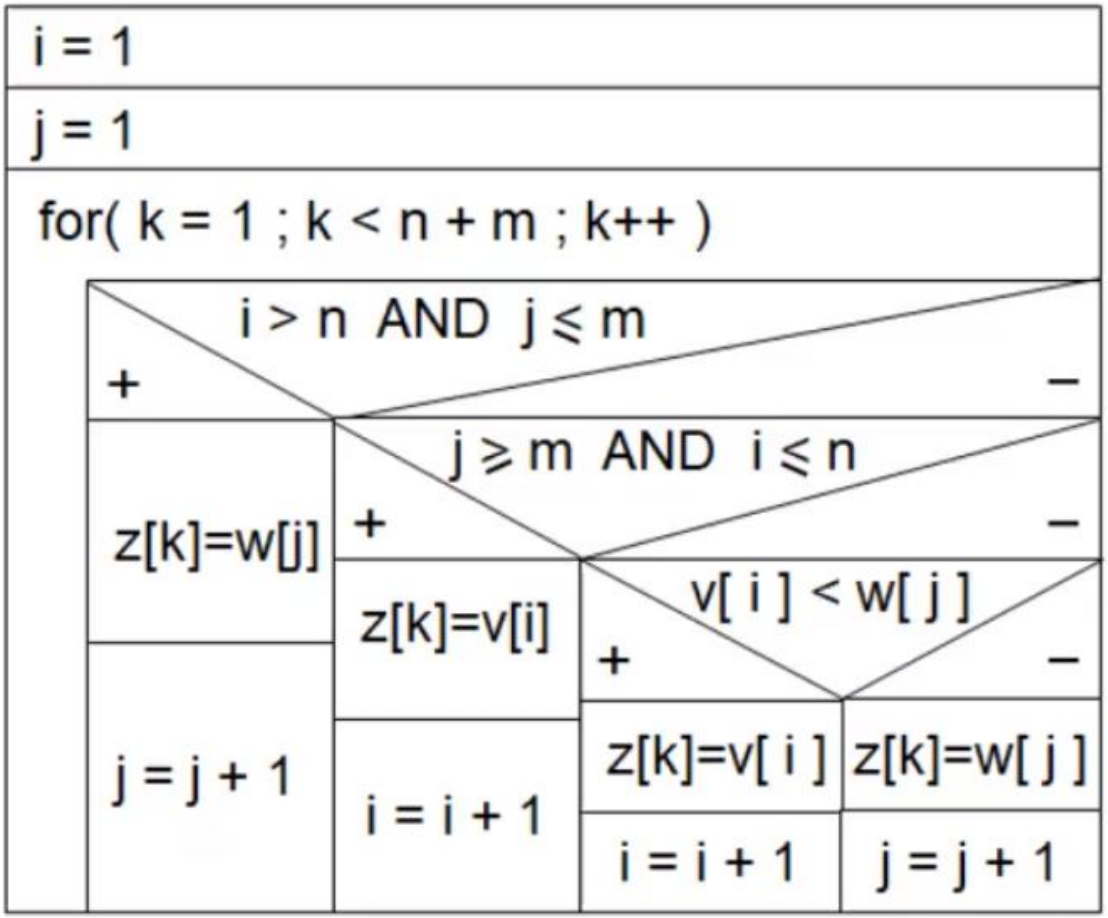
prototipo: `int[] merge (int[] v , int[] w)`
pre-condizione: `v` e `w` sono ordinati in modo crescente
post-condizione: restituisce un nuovo array ordinato in modo crescente con gli elementi di `v` e `w`



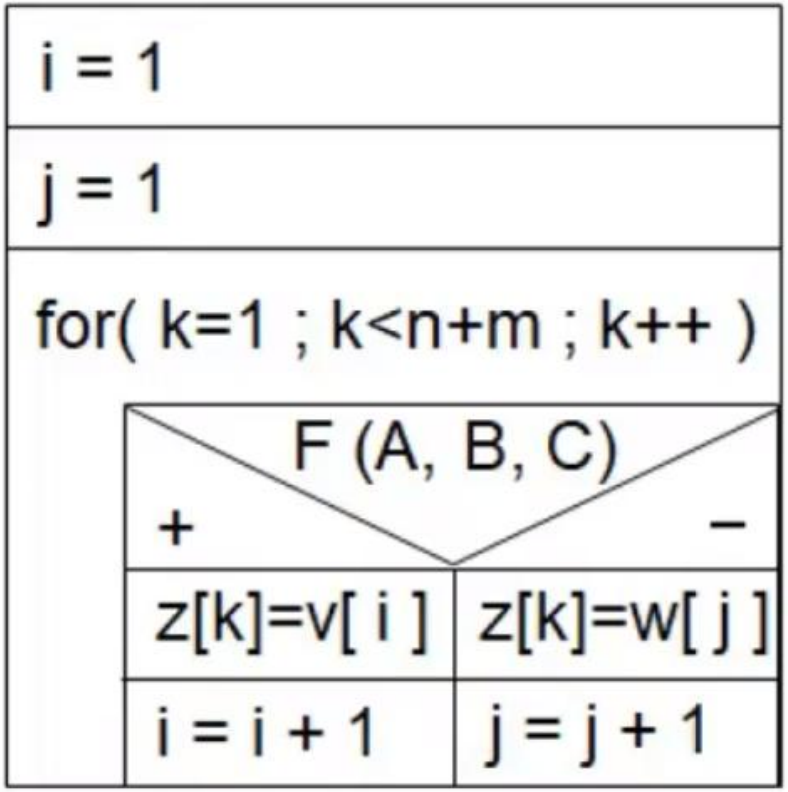
Esercizio


Inizio
i=1
j=1
k=1
j ≤ m
No
Sì
v[i]<w[k]
No
Sì
z[k]=w[j]
z[k]=v[i]
j=j+1
i=i+1
k=k+1
i ≤ n
Sì
No
j ≤ m
Sì
No
Fine

Esercizio


i = 1
j = 1
for(k = 1 ; k < n + m ; k++)
i > n AND j ⩽ m
+
−
z[k]=w[j]
j = j + 1
j ⩾ m AND i ⩽ n
+
−
z[k]=v[i]
i = i + 1
v[i] < w[j]
+
−
z[k]=v[i]
i = i + 1
z[k]=w[j]
j = j + 1

Esercizio



$$\begin{cases} A = & v[i] < w[j] \\ B = & i \leqslant n \\ C = & j \leqslant m \end{cases}$$

Esercizio

v[i]<w[j]	i≤n	j≤m	Situazione	Azione	F
A	B	C			
0	0	0	v e w sono terminati	caso impossibile	X
0	0	1	v è terminato	z[k]=w[j]	0
0	1	0	w è terminato	z[k]=v[i]	1
0	1	1	v[i]⩾w[j] i⩽n j ⩽m	z[k]=w[j]	0
1	0	0	v e w sono terminati	caso impossibile	X
1	0	1	v è terminato	z[k]=w[j]	0
1	1	0	w è terminato	z[k]=v[i]	1
1	1	1	v[i]<w[j] i⩽n j⩽m	z[k]=v[i]	1

ALGORITMO

Un algoritmo è un procedimento che risolve un determinato problema attraverso un numero finito di passi elementari, chiari e non ambigui.

Un algoritmo per essere eseguito su una macchina deve essere tradotto nel linguaggio che quella macchina riconosce.

L'algoritmo è quindi indipendente dal linguaggio di programmazione utilizzato.

Teorema di Böhm-Jacopini

Il teorema di Böhm-Jacopini, enunciato nel 1966 da due informatici italiani dai quale prende il nome, afferma che:

«qualunque algoritmo può essere implementato utilizzando tre sole strutture, la **sequenza**, la **selezione** e il **iterazione**, che possono essere tra loro innestati fino a giungere ad un qualsivoglia livello di profondità finito (come le scatole cinesi)»

In altre parole, qualsiasi procedimento risolutivo non strutturato («codice a spaghetti») può essere trasformato in un equivalente algoritmo strutturato.

DNS – Diagrammi Nassi-Shneiderman

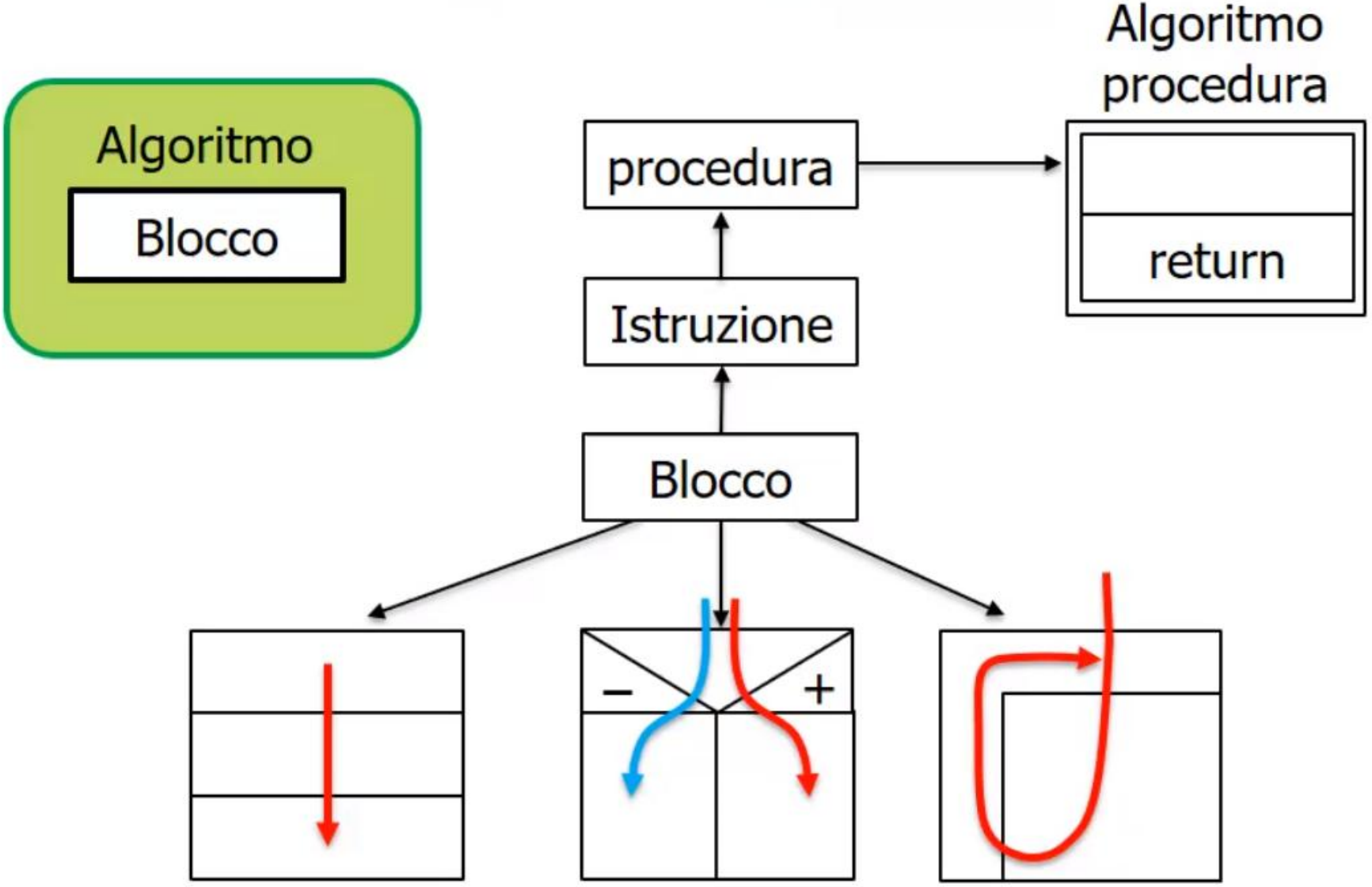
Isaac "Ike" Nassi
http://www.nassi.com/ike.html

Ben Shneiderman
http://www.cs.umd.edu/~ben/

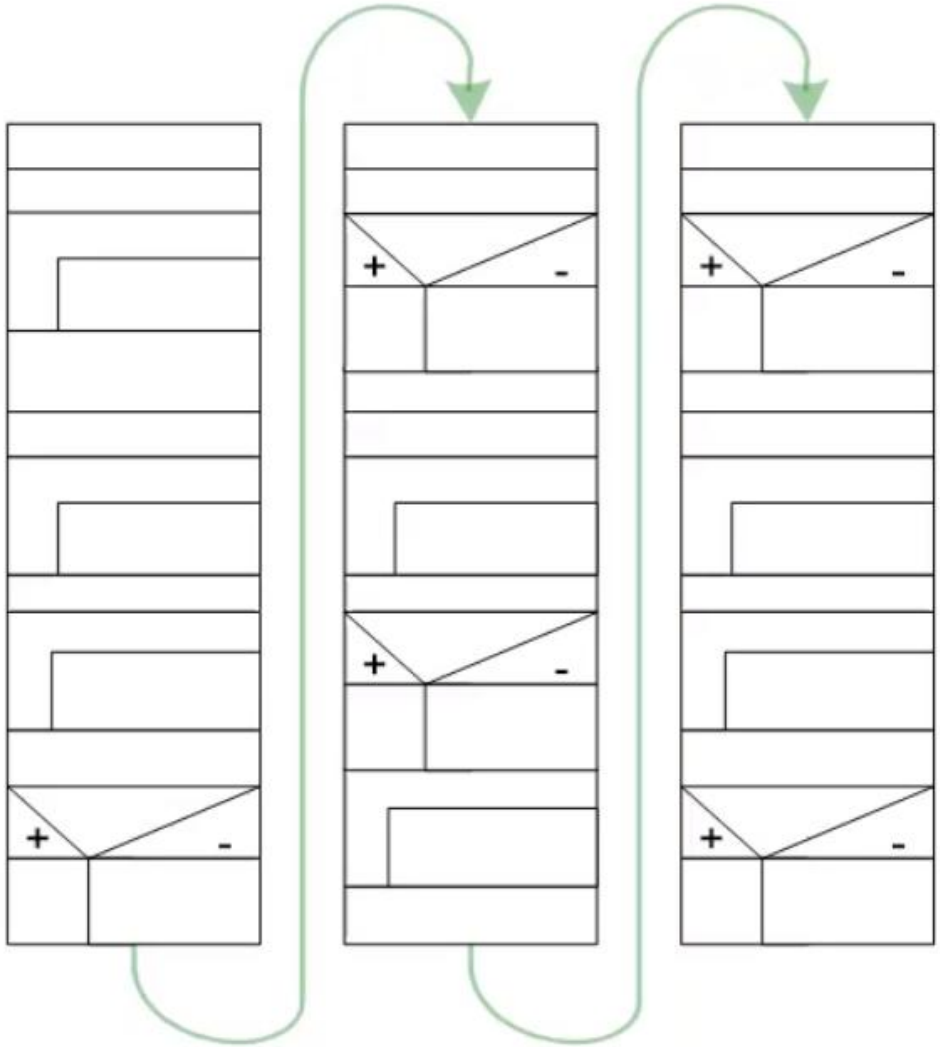
DNS – Diagrammi Nassi-Shneiderman

- sono un valido strumento didattico per scrivere codice strutturato
- hanno intrinsecamente il concetto di indentazione del codice quindi ne aumentano la leggibilità
- nascono con il linguaggio Pascal ma ne possono essere utilizzati in tutti i linguaggi procedurali (C++, java o assembly)
- si integrano perfettamente nelle metodologie di progettazione del software (es. UML) e costituiscono una valida alternativa ai diagrammi di flusso (che generano la programmazione "a spaghetti")
- la fase di traduzione nel linguaggio di programmazione è estremamente semplice

DNS – Definizione ricorsiva



Forma di un algoritmo DNS complesso

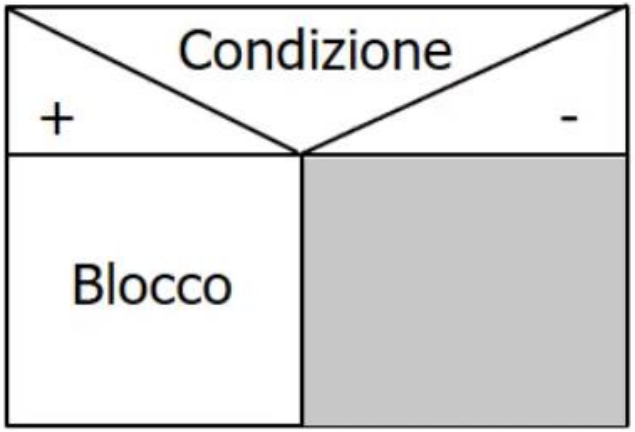


Blocchi di selezione

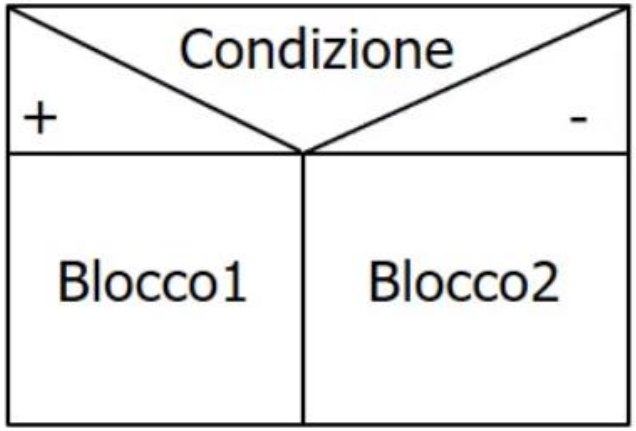
Selezionare significa poter intraprendere vie risolutive differenti a seconda di condizioni che sono verificate durante l'esecuzione. La selezione può essere:

- **Singola**: se la condizione è verificata (+) viene eseguito un blocco, altrimenti (-) si procede con l'algoritmo;
- **Binaria**: se la condizione è verificata (+) viene eseguito blocco, altrimenti (-) viene eseguito un secondo blocco;
- **Multipla**: controlla il valore di una variabile all'interno di un intervallo di valori predefiniti. Quando la variabile corrisponde a uno dei valori della selezione viene eseguito il blocco ad esso associato, altrimenti (se presente) il blocco di default.

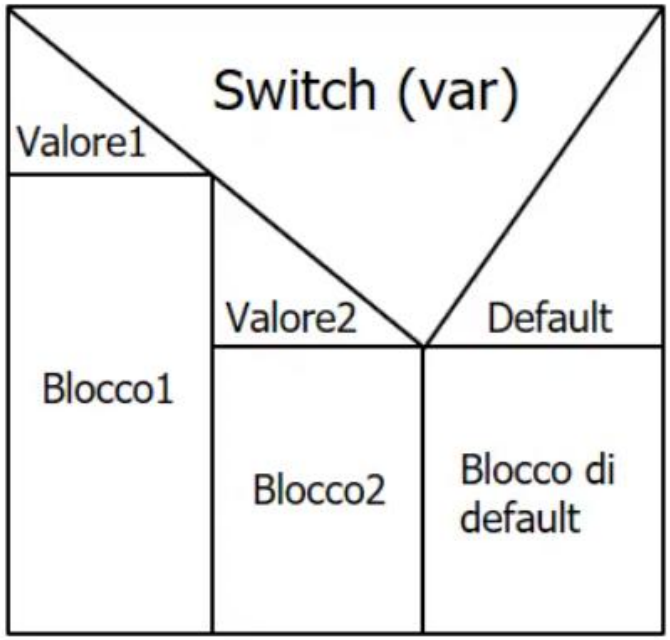
Blocco di selezione



Selezione singola



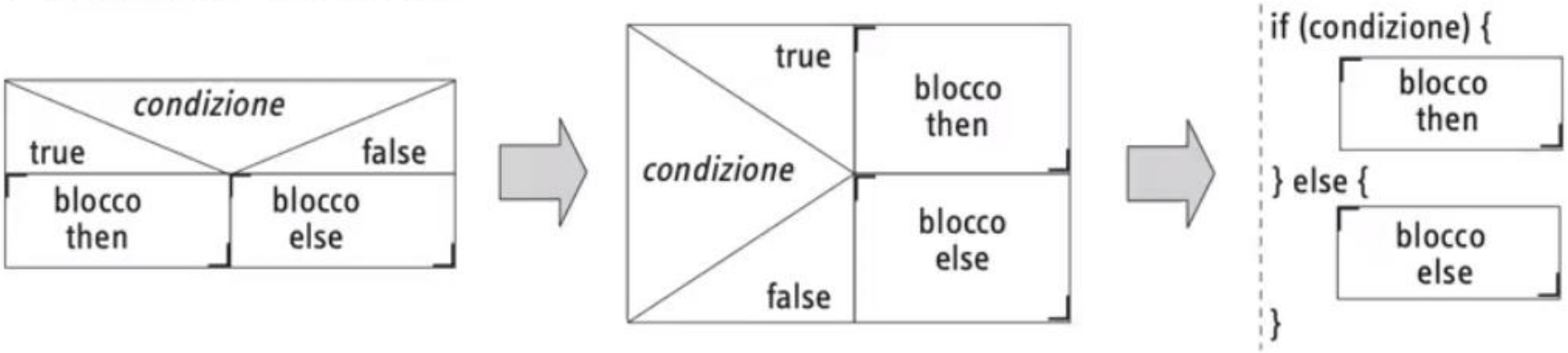
Selezione binaria



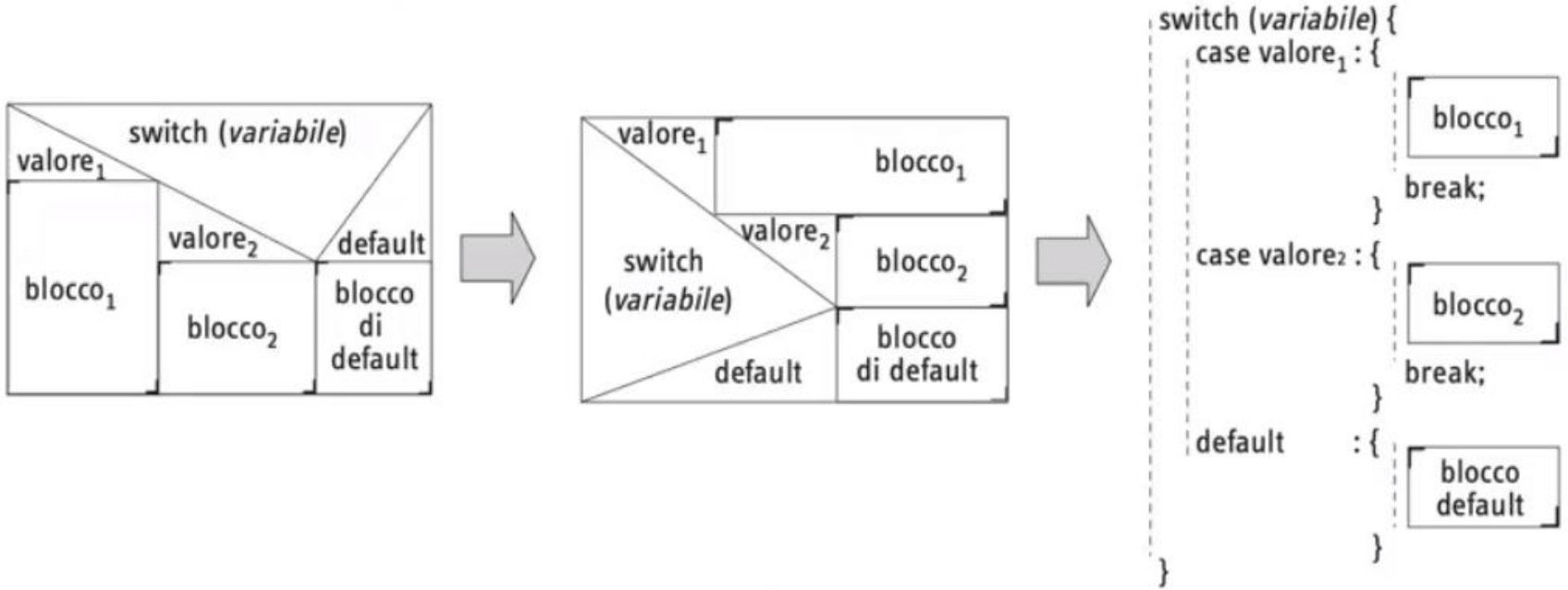
Selezione multipla

Trasformazione dei blocchi di selezione

Selezione binaria



Selezione multipla



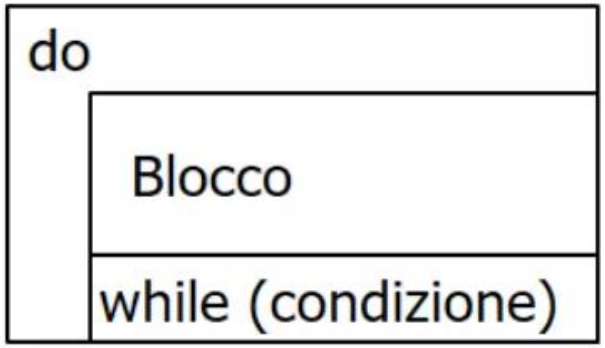
Blocchi iterativi

Un blocco di iterazione permette di eseguire ripetutamente un blocco di istruzioni fino a quando una condizione logica non è verificata.

Per via della loro natura ripetitiva un blocco di iterazione è anche detto **ciclo**.

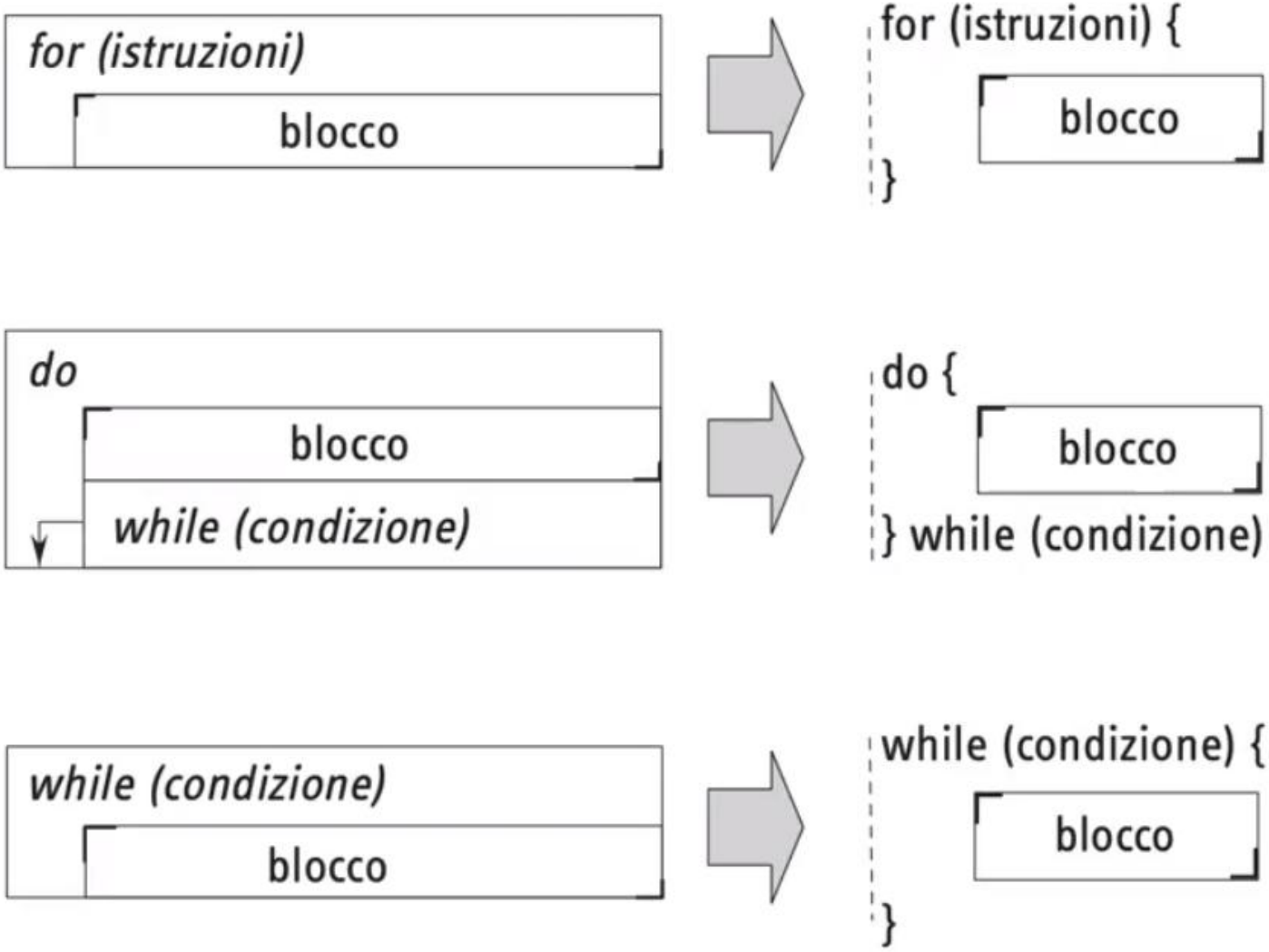
Ciclo for

Ciclo while



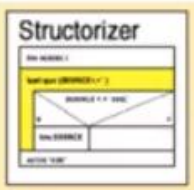
Ciclo do - while

Trasformazione dei blocchi di selezione



https://structorizer.fisch.lu

...traduce DNS in qualsiasi linguaggio di programmazione



STRUCTORIZER

- News
- Screenshots
- Downloads
- Android
- Changelog
- Bug Reports
- User Guide
- F.A.Q.
- Newsletter
- GDPR
- Login
- About

GPL v3

News

19.09.2021

The new comprehensive release 3.32 now came with the long-announced change to **require the Java 11 version** to run (3.31-04 was the last version satisfied with Java 8).
This also means that the Java WebStart installation opportunity does not work any longer - not at least directly. You might possibly try to adopt a workaround like the OpenWebStart if you want to adhere to it - JNLP files will continue to be provided from the Structorizer homepage. See the Downloads page for more details.

The most recent functional improvement is that you may display the Analyser warnings related to an element in a popup now, simply by having the mouse hover over the little red triangle in its corner.

(Further improvements are still on the To Do list, please stay tuned.)

see comments on this news

09.06.2021

Version 3.31-04 now offers substantially extended export opportunities to LaTeX - in addition to the StrukTeX diagram conversion Structorizer now also produces LaTeX pseudocode translations to four different packages:

- algorithmicx
- algorithmic (aka "algorithms")
- algorithm2e
- pseudocode

Enjoy it!

see comments on this news

01.03.2021

The new release 3.31 is on the brink. It enhances Structorizer by **two new source code parsers**:

- for Java (SE 8) and
- for Processing