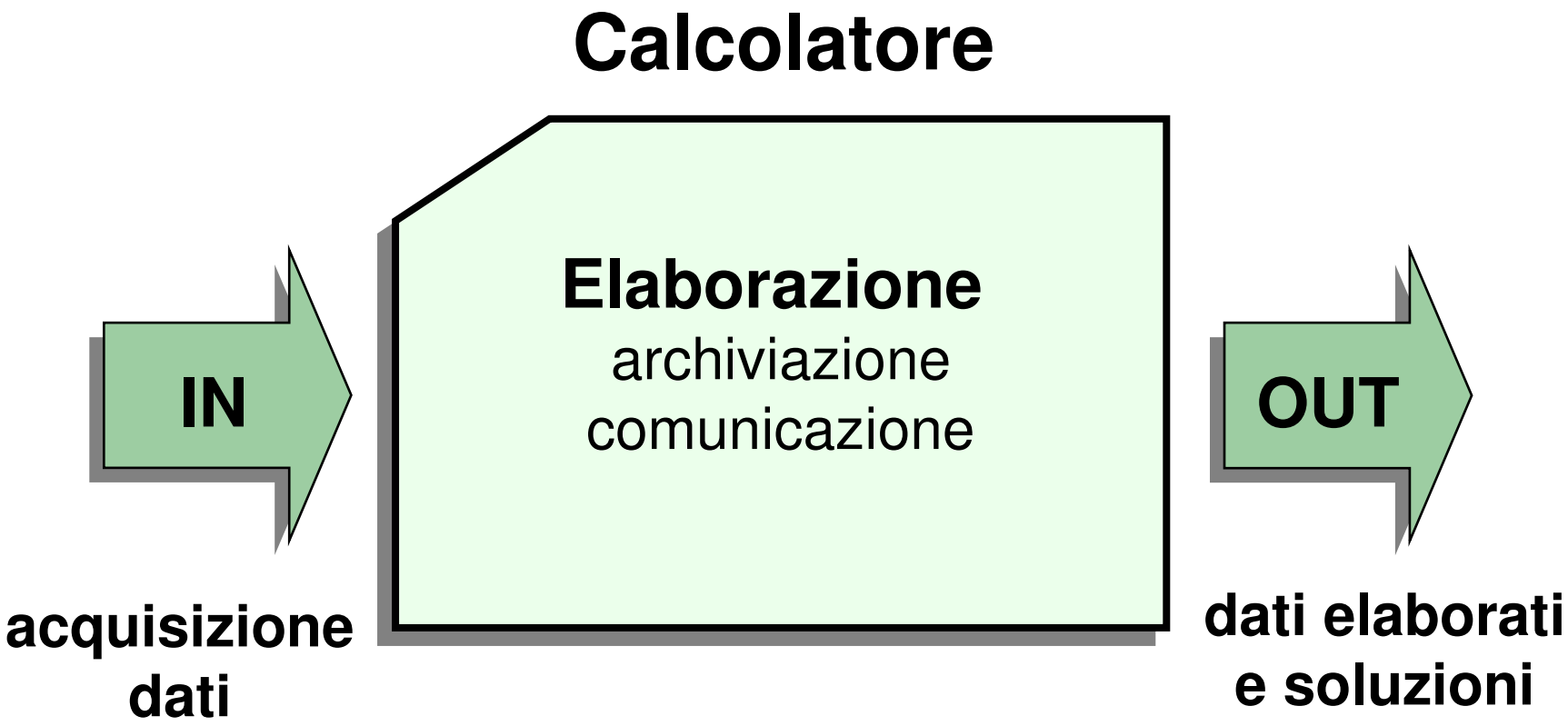
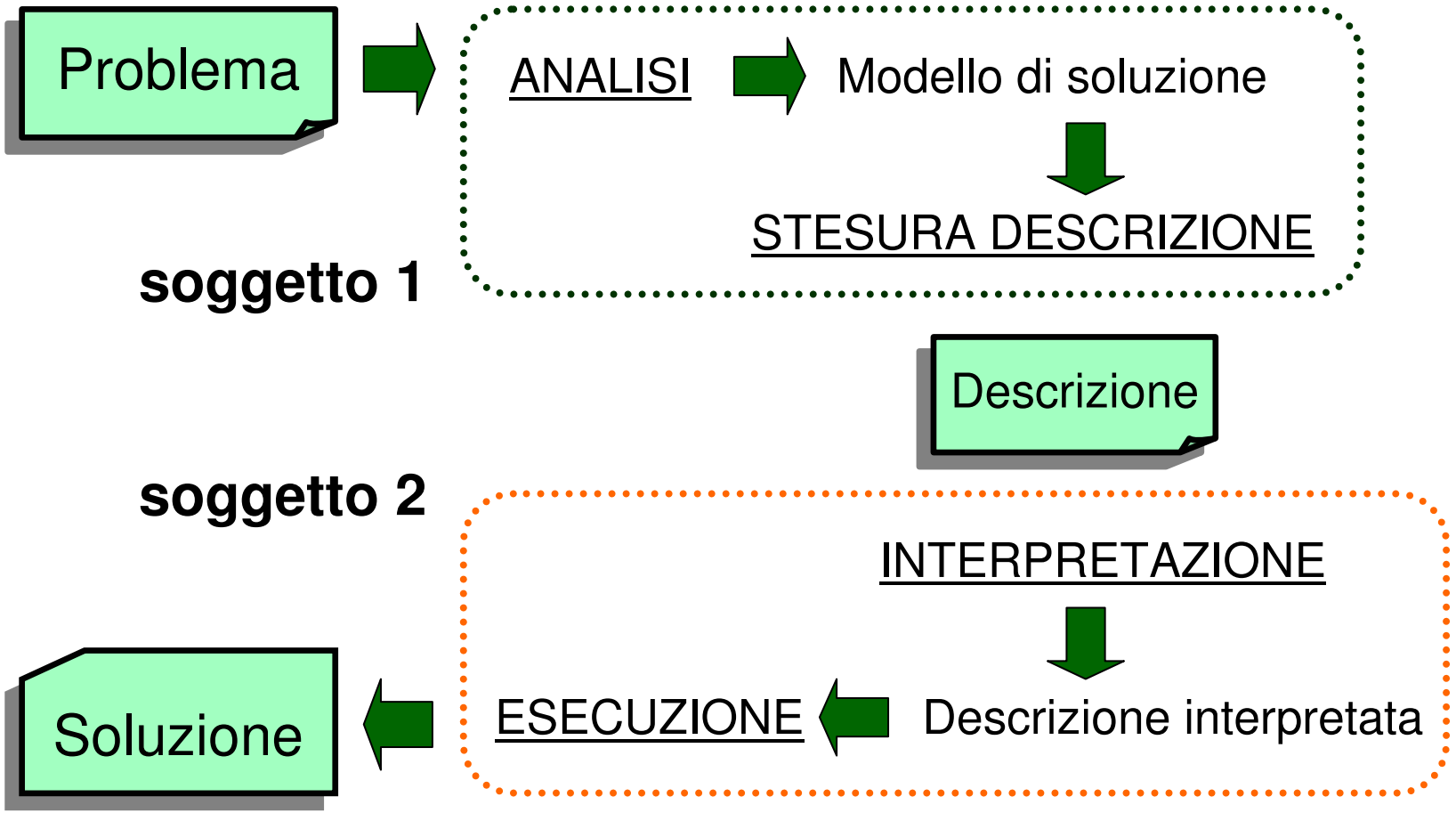
Parte Seconda: Trattamento dell'informazione

Fondamenti di informatica

Gestione dell'informazione



Soluzione di un problema

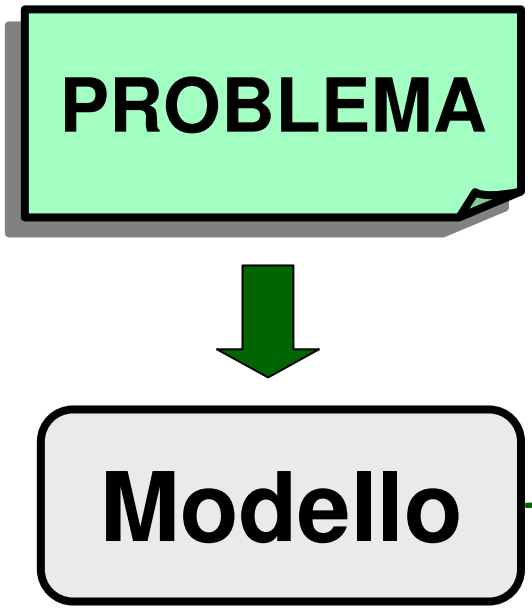


Affrontare il problema

L'**analisi** rappresenta un modo per passare dalla **specifica** di un problema alla sua **soluzione**, attraverso i seguenti passi:

- *Comprendere il problema,* ovvero eliminare le ambiguità di specifiche troppo informali ed individuare gli obiettivi prefissi
- *Modellare il problema*, ovvero fornire una rappresentazione semplificata della situazione da trattare e delle interazioni tra i diversi elementi in gioco
- *Ricercare una soluzione,* ovvero formulare una sequenza di operazioni la cui esecuzione porta alla soluzione del problema posto.

Modellare il problema



modello grafico
modello tabellare
modello simbolico

Rappresentazione semplificata della situazione che descrive, attraverso elementi, loro proprietà e relazioni tra di essi.

Un problema ed una sua soluzione

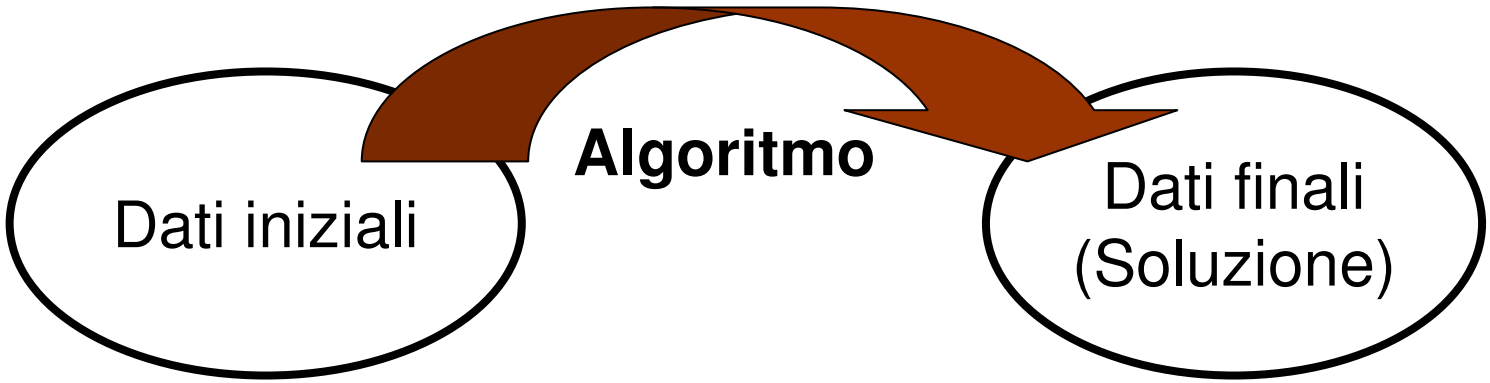
- **Problema: esame universitario.**
- **Soluzione:**

0. Decidi quale esame sostenere
1. Procurati il materiale di studio
2. Frequenta il corso relativo all'esame scelto
3. **Studia**
4. La preparazione non sembra sufficiente? Torna al punto 3, altrimenti:
5. Iscriviti all'esame
6. Sostieni l'esame
7. Ad esame non superato o valutazione rifiutata torna al punto 3, altrimenti **fine.**

Algoritmi e linguaggi

Definizione di Algoritmo

Si definisce **algoritmo** una sequenza di azioni che trasformi i dati iniziali in un numero *finito* di passi, elementari e non ambigui, per giungere al risultato finale.

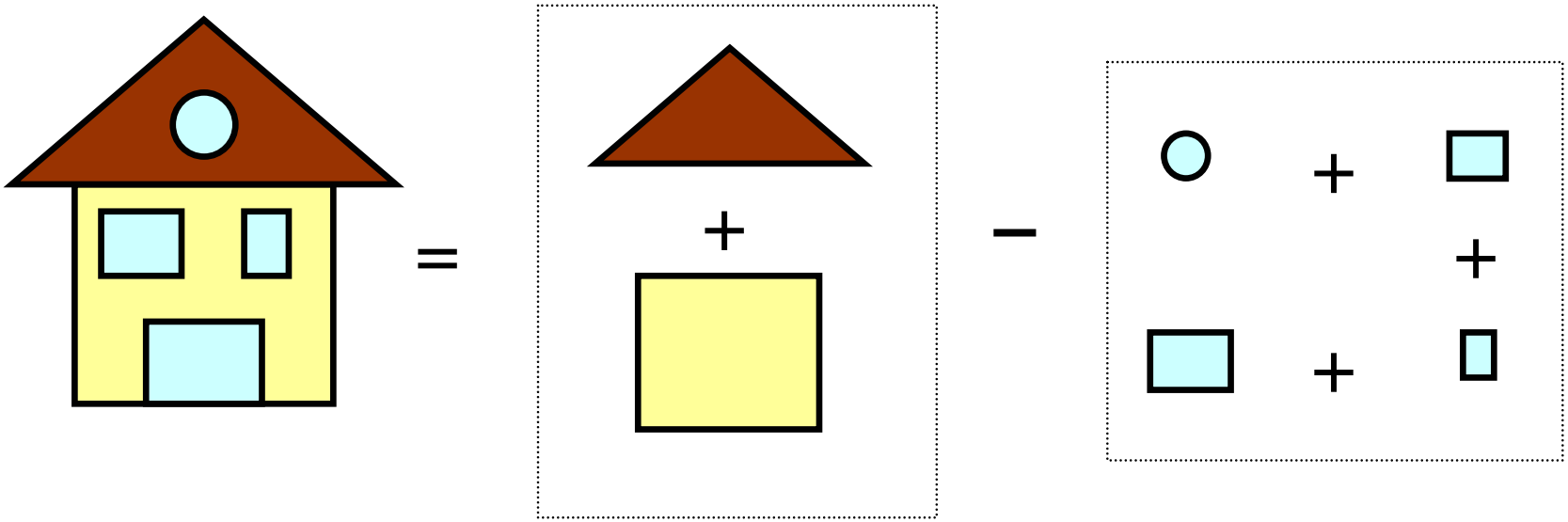


Operazioni non elementari

- Tutte le operazioni specificate dall'algoritmo devono essere eseguibili dall'esecutore **(operazioni elementari)**
- In caso di operazioni complesse è necessario "scomporre" il problema in **sottoproblemi** più semplici
  - Nell'esempio dell'uovo al burro l'operazione di rottura del guscio potrebbe richiedere una spiegazione più approfondita (sottoproblema).

Esempio: area di una “casa”

Il calcolo dell’area della “facciata” si può ricondurre al problema più semplice del calcolo delle aree dei singoli componenti, calcolando l’area totale della “muratura” e sottraendo l’area totale di porte e finestre.



Esempio: gestione biblioteca

- Insieme di libri disposti sugli scaffali
- La posizione di ogni libro è individuata da due coordinate:
  - **Scaffale** (numero dello scaffale)
  - **Posizione** nello scaffale
- La biblioteca è dotata di uno schedario (ordinato per autore/i e titolo). Ogni scheda contiene, nell'ordine:
  - **cognome** e **nome** dell'autore
  - **titolo** del libro
  - **data** di pubblicazione
  - **numero dello scaffale** in cui si trova
  - **posizione** del libro nello scaffale.

Una scheda

Autore/i: Pasquali, Giulio
Principe, Remy

Titolo: **Il Violino**,
III Edizione, 1951

Scaffale: 16

Posizione: 9

Formulazione dell'algoritmo

Decidi che libro richiedere

Preleva il libro richiesto.

Se un passo dell'algoritmo non è direttamente comprensibile ed eseguibile, occorre dettagliarlo a sua volta mediante un ulteriore algoritmo.

Un algoritmo di prelievo

1. Decidi che libro richiedere
2. Cerca la scheda del libro richiesto
3. Segnati numero scaffale e posizione
4. Cerca lo scaffale indicato
5. Accedi alla posizione indicata e preleva il libro
6. Scrivi i tuoi dati sulla “scheda di prestito”.

Un “sotto-algoritmo”

1. Prendi la prima scheda
2. Esamina se titolo e autore/i sono quelli cercati.
   In caso positivo la ricerca termina con successo,
   altrimenti passa alla scheda successiva e ripeti
3. Se esaurisci le schede il libro cercato non esiste o
   non è nella biblioteca.

Problema: cosa succede se l’autore cercato è “Zavattini”?

Un “sotto-algoritmo” migliore

1. Esamina la scheda centrale dello schedario
2. Se la scheda centrale corrisponde al libro cercato allora termina
3. Altrimenti, prosegui allo stesso modo nella metà superiore o inferiore dello schedario a seconda che il libro segua o preceda quello indicato sulla scheda.

Problema: l’algoritmo è incompleto: c’è un’altra condizione di terminazione quando il libro non esiste.

Revisione del passo 2

Se la scheda centrale corrisponde al libro cercato oppure se la parte di schedario da consultare è vuota, allora termina.

Libro trovato

Libro inesistente o non in archivio

Esempio: algoritmo sequenziale

1. Alzarsi dal letto
2. Togliersi il pigiama
3. Fare la doccia
4. Vestirsi
5. Fare colazione
6. Uscire in bicicletta

N.B. I passi sono eseguiti in sequenza e l'**ordine** delle istruzioni è essenziale per la correttezza dell'algoritmo.

Strutture condizionali

1. Alzarsi dal letto
2. Togliersi il pigiama
3. Fare la doccia
4. Vestirsi
5. Fare colazione
6. **Se** piove **allora** prendere l'ombrello
7. Uscire in bicicletta.

Se…altrimenti...

1. Alzarsi dal letto
2. Togliersi il pigiama
3. Fare la doccia
4. Vestirsi
5. Fare colazione
6. **Se** piove **allora** prendere l'auto
   altrimenti prendere la bicicletta.

Ciclo “mentre”

1. Alzarsi dal letto
2. Togliersi il pigiama
3. Fare la doccia
4. Vestirsi
5. Fare colazione
6. **Mentre** piove, restare in casa
7. Uscire in bicicletta.

Strutture di controllo

> Gli esempi precedenti evidenziano diverse *strutture di controllo:*

- **sequenziale:** le istruzioni vengono semplicemente effettuate una dopo l'altra.
- **condizionale:** le istruzioni da eseguire sono determinate dalla valutazione di una determinata *condizione* (***se***)
- **Iterativa:** le istruzioni devono essere eseguite ripetutamente fino a che non si verifica una determinata condizione (**mentre, ripeti**).

Definizione informale di Algoritmo

"Sequenza **finita** di istruzioni **comprensibili** da un **esecutore** che descrive come **realizzare un compito**, ovvero come **risolvere un problema.**"

Esempi:

- Istruzioni di montaggio di un elettrodomestico
- Uso di un terminale Bancomat
- Calcolo del massimo comun divisore di naturali.

Linguaggi umani ed informatici

- Il linguaggio **umano** è quello che tutti noi adottiamo per comunicare e scrivere; è *informale* e spesso *ambiguo*
- Il linguaggio **informatico** è la "lingua" dei computer; è *formale* e *non ambiguo*
- Gli algoritmi presentati finora sono scritti in linguaggio umano, e non possono essere "capiti" da un esecutore automatico.

Esecutori e linguaggi

- Un **esecutore** è definito in base a tre elementi:
  - l'insieme di **operazioni** che è in grado di compiere;
  - l'insieme delle **istruzioni** che capisce **(sintassi);**
  - quali operazioni **associa** ad ogni **istruzione** che riconosce **(semantica)**
- Il calcolatore capisce le istruzioni che fanno parte del **linguaggio macchina** (linguaggio informatico).

Soluzione effettiva per l'esecutore

- Un problema è **semplice** per un esecutore se l'esecutore può svolgerlo direttamente.
- Altrimenti il *descrittore* deve scomporre il problema in sottoproblemi finché l'algoritmo non è espresso esclusivamente tramite **operazioni elementari**
- Una soluzione è **effettiva** se l'esecutore è in grado di
  - interpretarla
  - compiere le azioni in tempo **finito.**

Proprietà dell'azione elementare

- **Finitezza:** l'azione deve concludersi in tempo finito
- **Osservabilità:** gli effetti dell'azione devono essere "visibili", ovvero devono produrre qualcosa
- **Riproducibilità:** a partire dallo stesso stato iniziale, la stessa azione deve produrre sempre lo stesso risultato.

Linguaggi di programmazione

- Rappresentano la “lingua” per descrivere gli algoritmi
- Un **programma** è la descrizione di un algoritmo mediante un determinato linguaggio di programmazione
- Un linguaggio di programmazione è costituito da un insieme di **parole chiave** (keywords)
- Un linguaggio è identificato da una **sintassi** e da una **semantica.**

Linguaggio macchina

- Rappresenta l'unico linguaggio **direttamente eseguibile** dall'elaboratore
- Non necessita quindi di una "traduzione" ulteriore
- La sua rappresentazione è la codifica binaria
- Risulta di difficile comprensione da parte dell'utente, che solitamente programma in linguaggi di **alto livello.**

Linguaggi di alto livello

- Ogni istruzione di questi linguaggi esprime una serie di azioni elementari
- L'esecuzione di un programma scritto in linguaggio di alto livello necessita una traduzione in linguaggio macchina **(compilazione)**
- Questi linguaggi **astraggono** dai dettagli legati all'architettura e sono di **semplice comprensione.**

Dal problema al programma

- **Specifiche dei requisiti:** descrizione precisa e **corretta** dei requisiti
  - **“cosa?”**
- **Progetto:** procedimento con cui si individua la soluzione
  - **“come?”**
- **Soluzione: algoritmo.**

Proprietà degli algoritmi

- **Correttezza:** l'algoritmo perviene alla soluzione del compito cui è preposto, senza difettare di alcun passo fondamentale
- **Efficienza:** l'algoritmo perviene alla soluzione del problema usando la minima quantità di risorse fisiche (tempo, memoria, ecc.).

Proprietà di *un* algoritmo

- **Univocità:** non deve esistere alcun grado di libertà da parte del processore nell'esecuzione di ogni azione
- **Effettività:** le operazioni prescritte dall'algoritmo devono poter essere eseguite in *tempo finito*
- **Ingresso:** per ogni possibile scelta dei dati in ingresso l'algoritmo fornisce un risultato coerente
- **Uscita:** l'algoritmo deve fornire uno o più dati in uscita
- **Terminazione:** l'esecuzione di un algoritmo deve terminare in un numero *finito* di passi.

Riepilogo

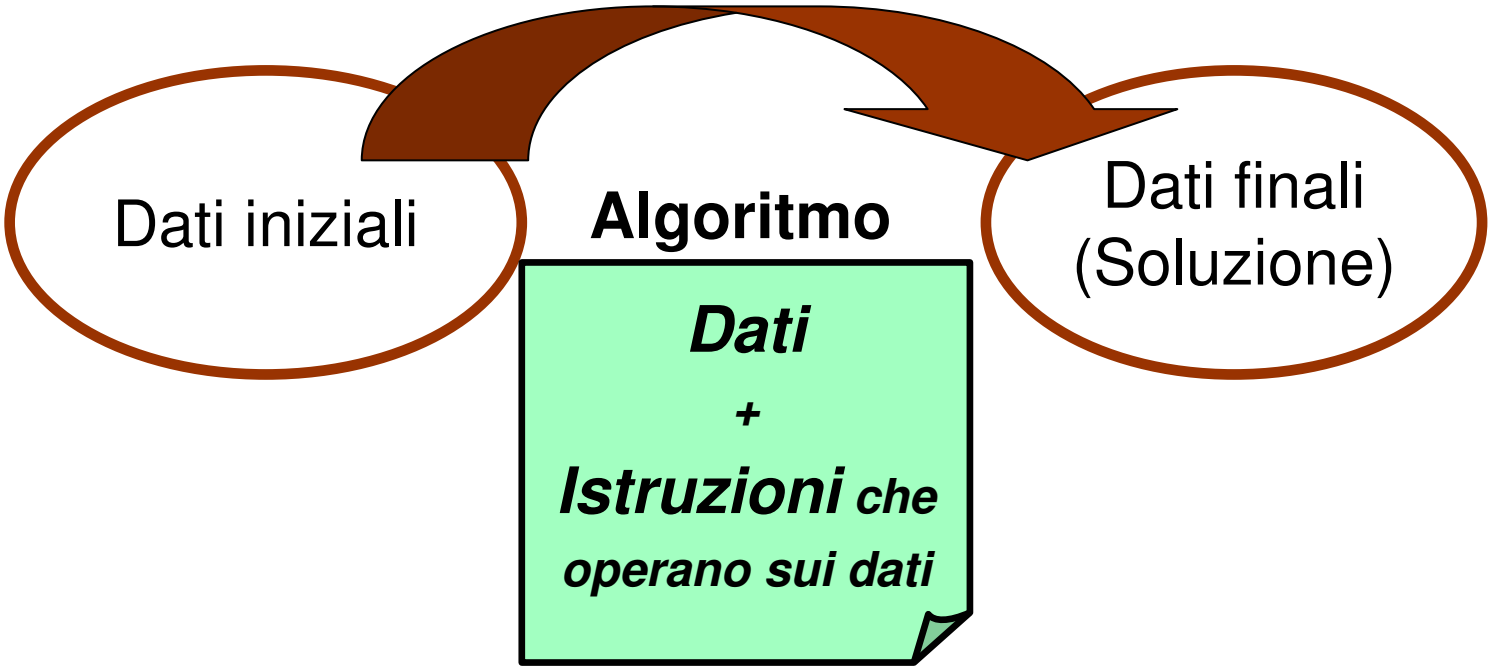
- **Algoritmo:** descrizione di come si risolve un problema.
- **Programma:** algoritmo scritto in modo che possa essere eseguito da un calcolatore *(linguaggio di programmazione)*
- **Linguaggio macchina:** linguaggio effettivamente "compreso" da un calcolatore, caratterizzato da:
  - istruzioni primitive semplici ed efficienti
  - difficile e noioso da usare per un programmatore
- Il **programmatore** *produce algoritmi,* e li *codifica in programmi* comprensibili al calcolatore.

Codifica degli algoritmi

> Algoritmo in linguaggio umano
  - sintetico ed intuitivo
  - codificato in linguaggi informali o semi-formali (linguaggio naturale, diagrammi di flusso, ecc.)

> Algoritmo in linguaggio informatico
  - preciso ed eseguibile da un calcolatore
  - codificato in linguaggi di programmazione (linguaggio macchina o ad alto livello).

Algoritmo = dati + istruzioni



Dati e istruzioni

> **Tipi di dati**
  - Numeri naturali, interi o reali — 16, -9, 0.77 ..
  - Caratteri alfanumerici — a, b, .. , A, B, ..
  - Stringhe — “Turing”, “Mozart”..
  - Dati logici o booleani — Vero, Falso
  - Array di n elementi (omogenei) — {0,6,8,4,1,19}
  - Record (disomogenei) — [“pi greco”, 3.14159]

> **Istruzioni**
  - Operazioni di Input/Output — *leggi, scrivi ..*
  - *Operazioni Aritmetico-logiche* — max = A + B ..
  - Strutture di controllo — *mentre, ripeti..*

Algebra Booleana

- L'algebra Booleana (dal suo inventore G. Boole) serve a descrivere le operazioni logiche.
- L'algebra si compone di:
  - operatori booleani
  - regole di trasformazione ed equivalenza tra operatori.
- Gli **operandi** booleani assumono solo **due** valori:

  Vero / Falso True / False 1 / 0 Si / No ..

Operatori e tavole di verità

A	not A
0	1
1	0

Negazione

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

Congiunzione

A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

Disgiunzione

A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

Disgiunzione esclusiva

A	B	A $\leftrightarrow$ B
0	0	1
0	1	0
1	0	0
1	1	1

Doppia implicazione

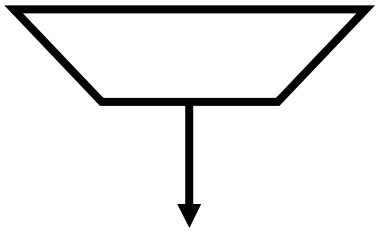
Operazioni elementari

- Operazioni aritmetiche e assegnamento di valori a singole **variabili**
  - es. somma = (A + B)
- Condizioni sul valore di singole variabili
  - **se** (A > B) **allora … altrimenti …**
- Lettura e scrittura di variabili
  - “leggi A” … “stampa B” ...

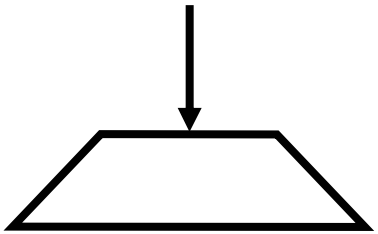
Rappresentazione degli algoritmi

- Linguaggio naturale
- Diagramma di flusso
- Pseudo codice
- Linguaggio di programmazione

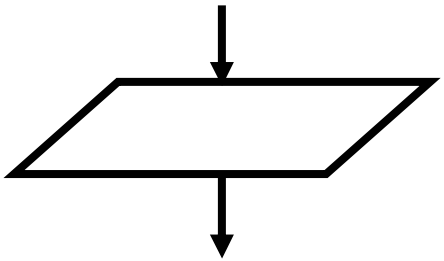
Diagrammi di flusso



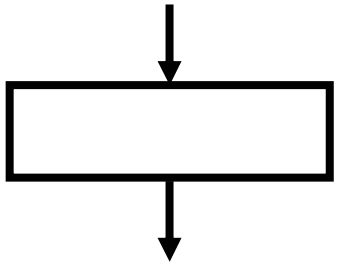
Inizio



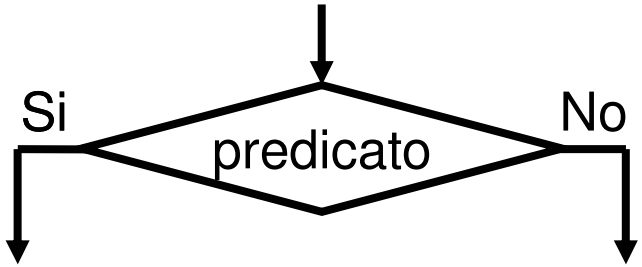
Fine



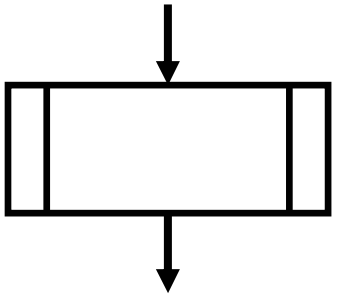
Input/Output



Elaborazione



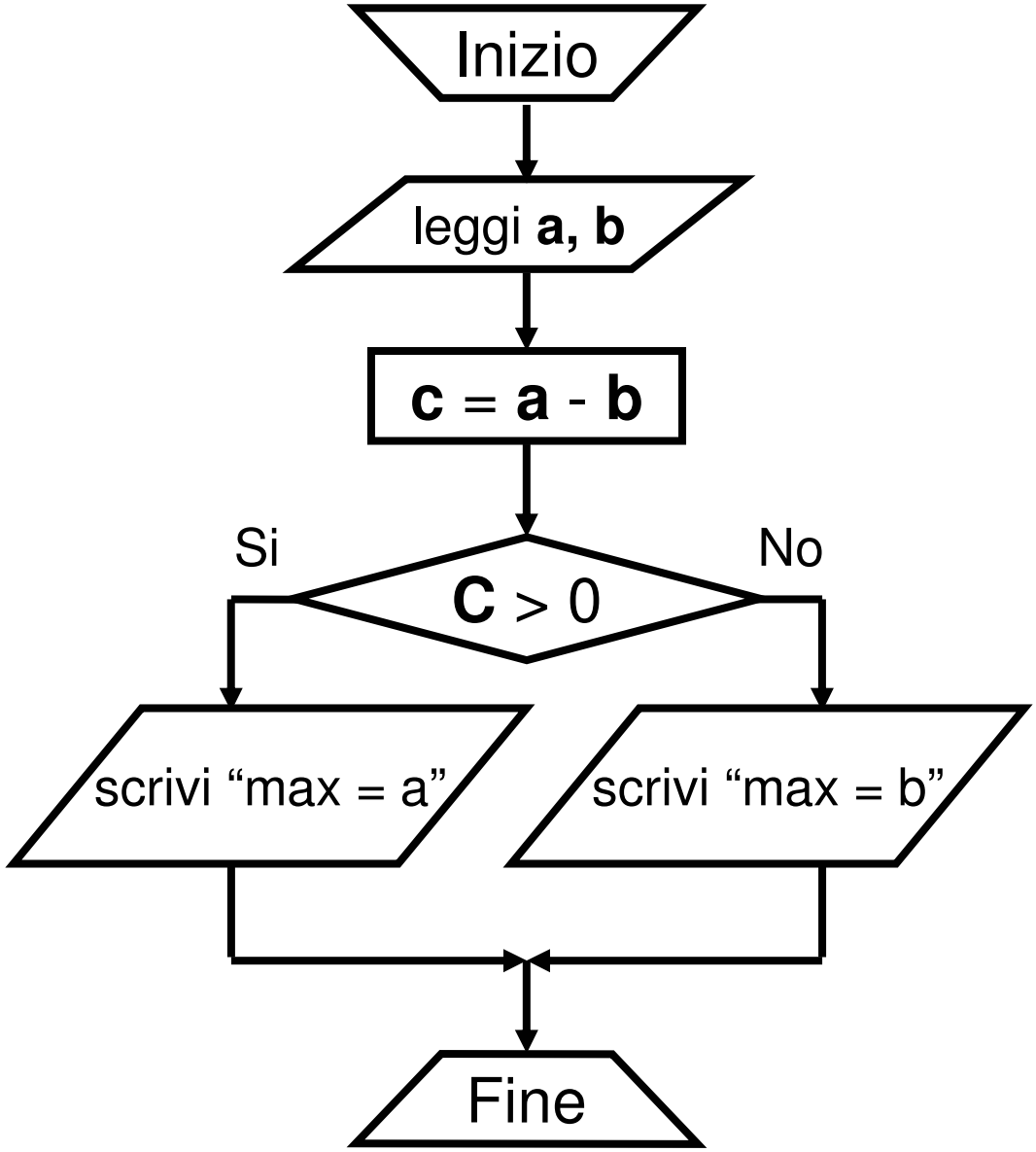
Selezione a due vie



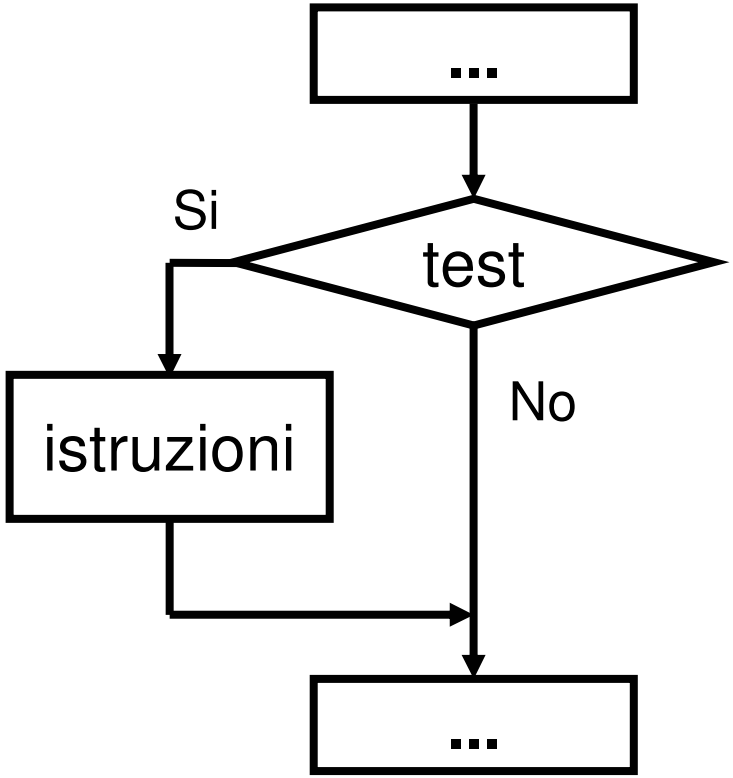
Sottoprogramma

Esempio

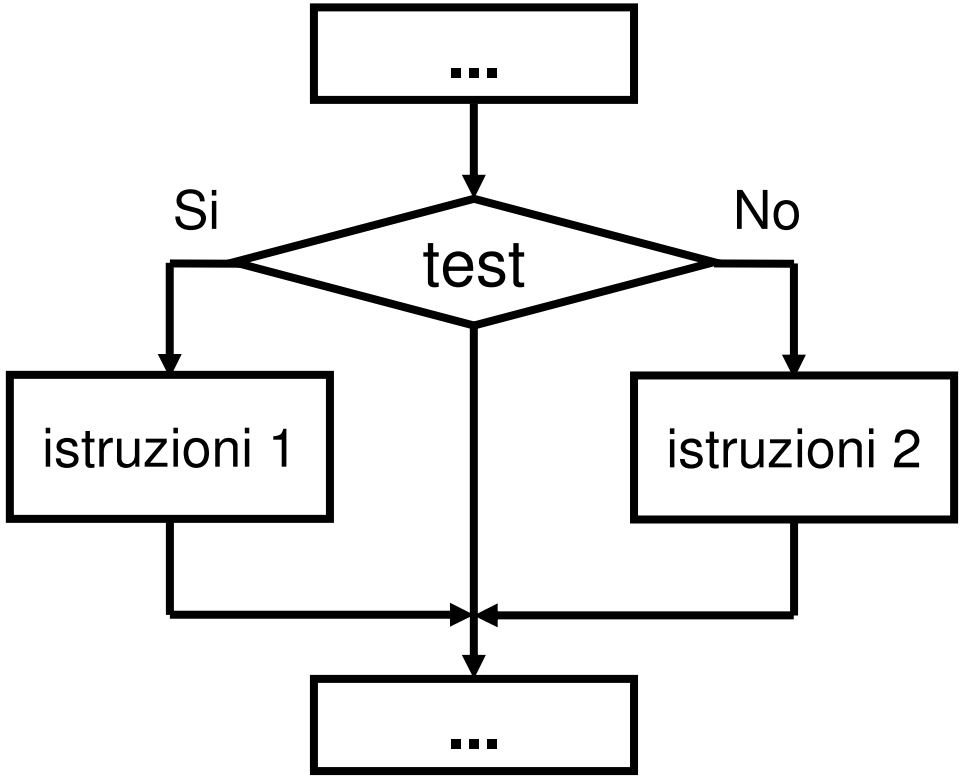
"Dati due numeri **a** e **b**, si calcoli e stampi il maggiore"



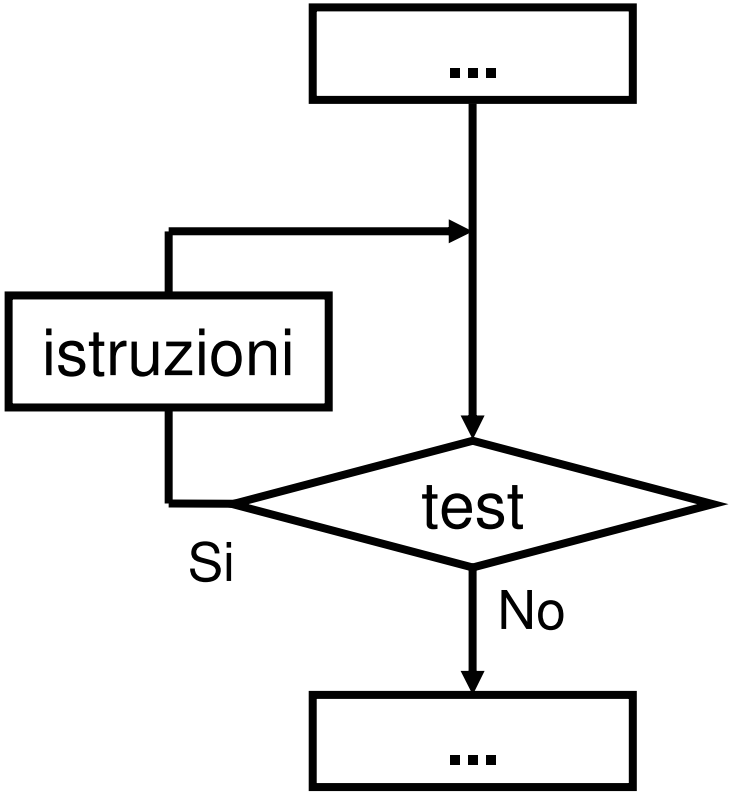
Selezione semplice



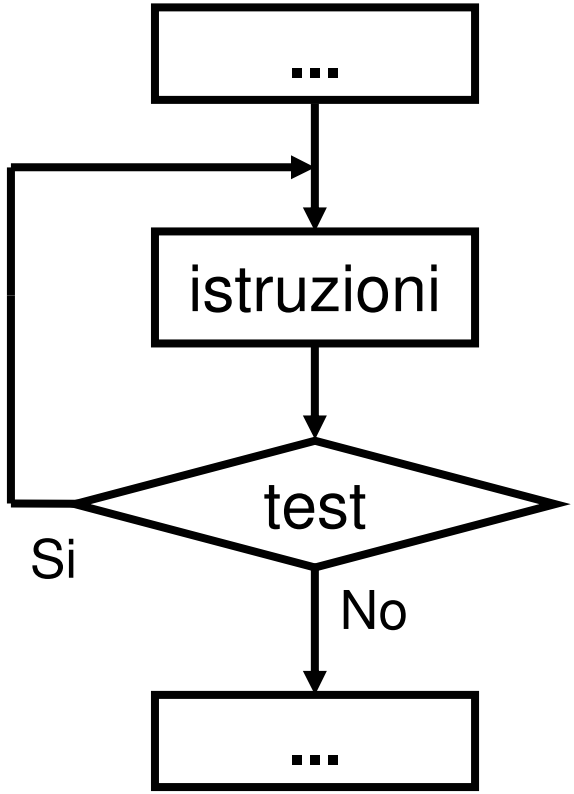
Selezione a due vie



Ciclo a condizione iniziale



Ciclo a condizione finale



Esempi di algoritmo

Il prodotto di due interi positivi

Algoritmo in **linguaggio naturale**

- Leggi W
- Leggi Y
- Somma W a se stesso Y volte
- Scrivi il risultato

Il prodotto di due interi positivi

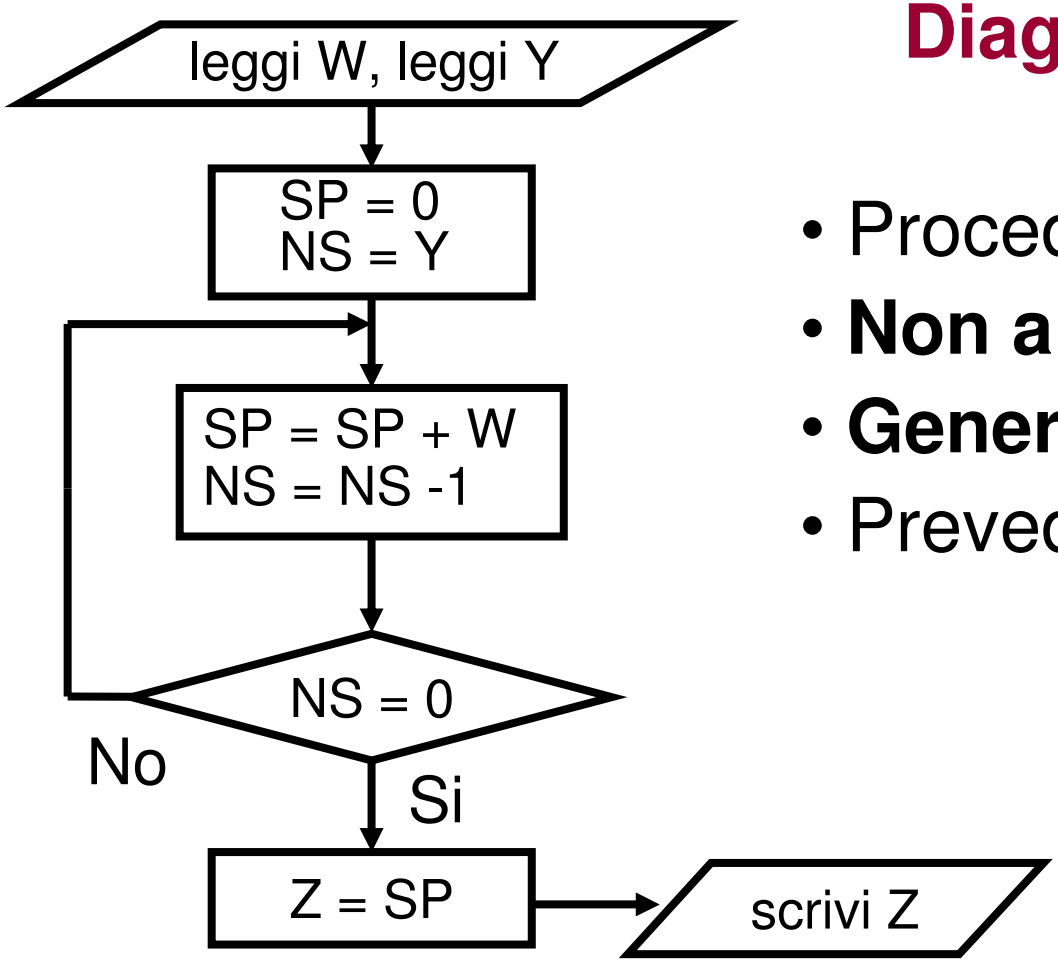
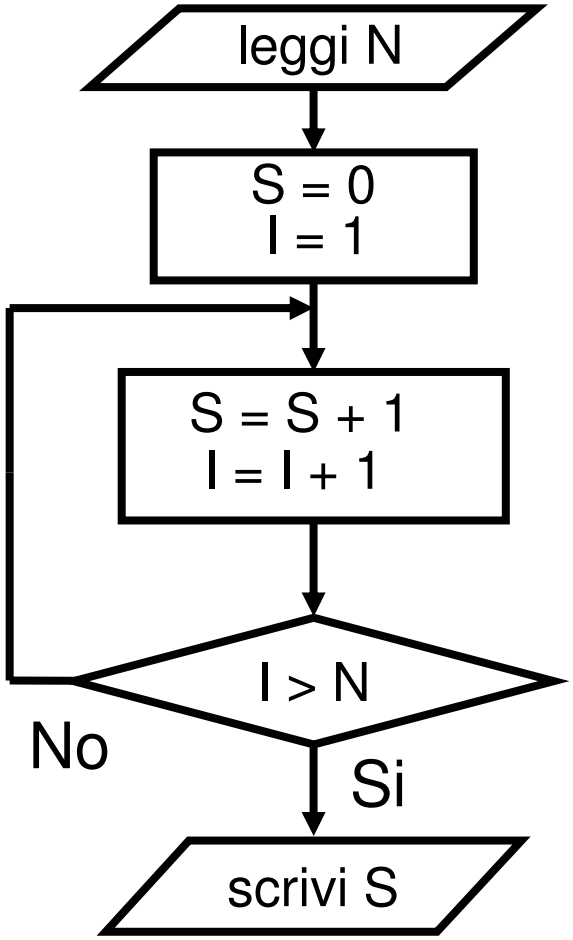


Diagramma di flusso

- Procedimento **sequenziale**
- **Non ambiguo**
- **Generale**
- Prevede **tutti i casi.**

Esempio



Calcolare e stampare la somma dei primi N numeri naturali.

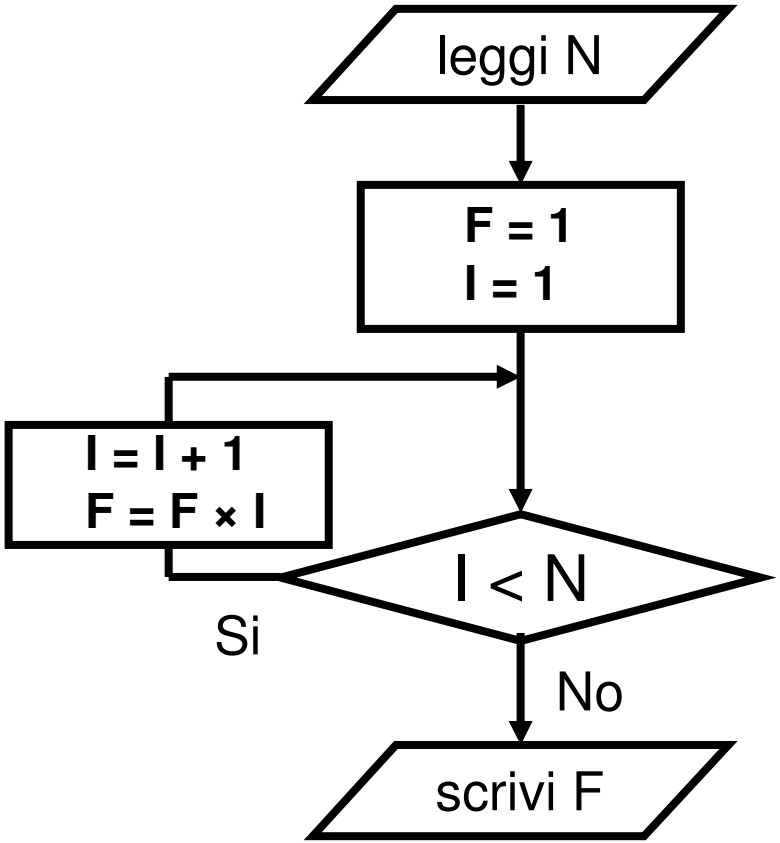
Esercizio

- L'esecutore deve leggere un intero N e restituire il **fattoriale** di questo numero, ovvero il valore:

  N x (N - 1) x (N -2) X … X 1.

- Scrivere l'algoritmo assumendo che i dati in ingresso siano sempre corretti (N > 0)
- Modificare poi l'algoritmo in modo da considerare la possibilità di valori di N < 1.

Una possibile soluzione



Stato dei dati (N = 7)

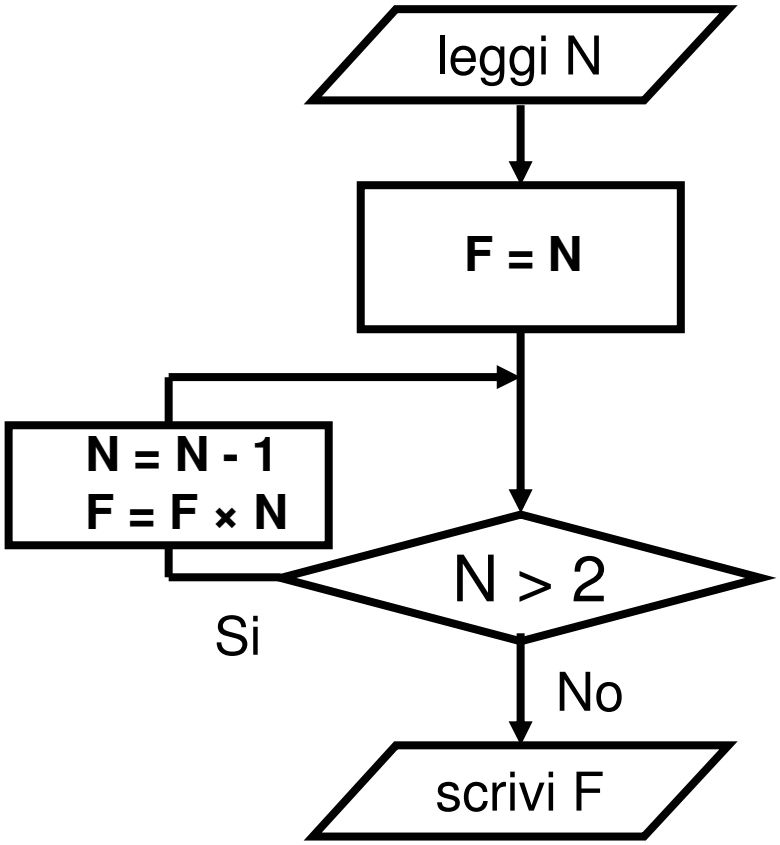
F		I		F
1	×	2	=	2
2	×	3	=	6
6	×	4	=	24
24	×	5	=	120
120	×	6	=	720
720	×	7	=	**5040**

Casi particolari:

N = 0 produce F = 1

N = -4 Produce F = 1

Un’altra soluzione



Stato dei dati (N = 7)

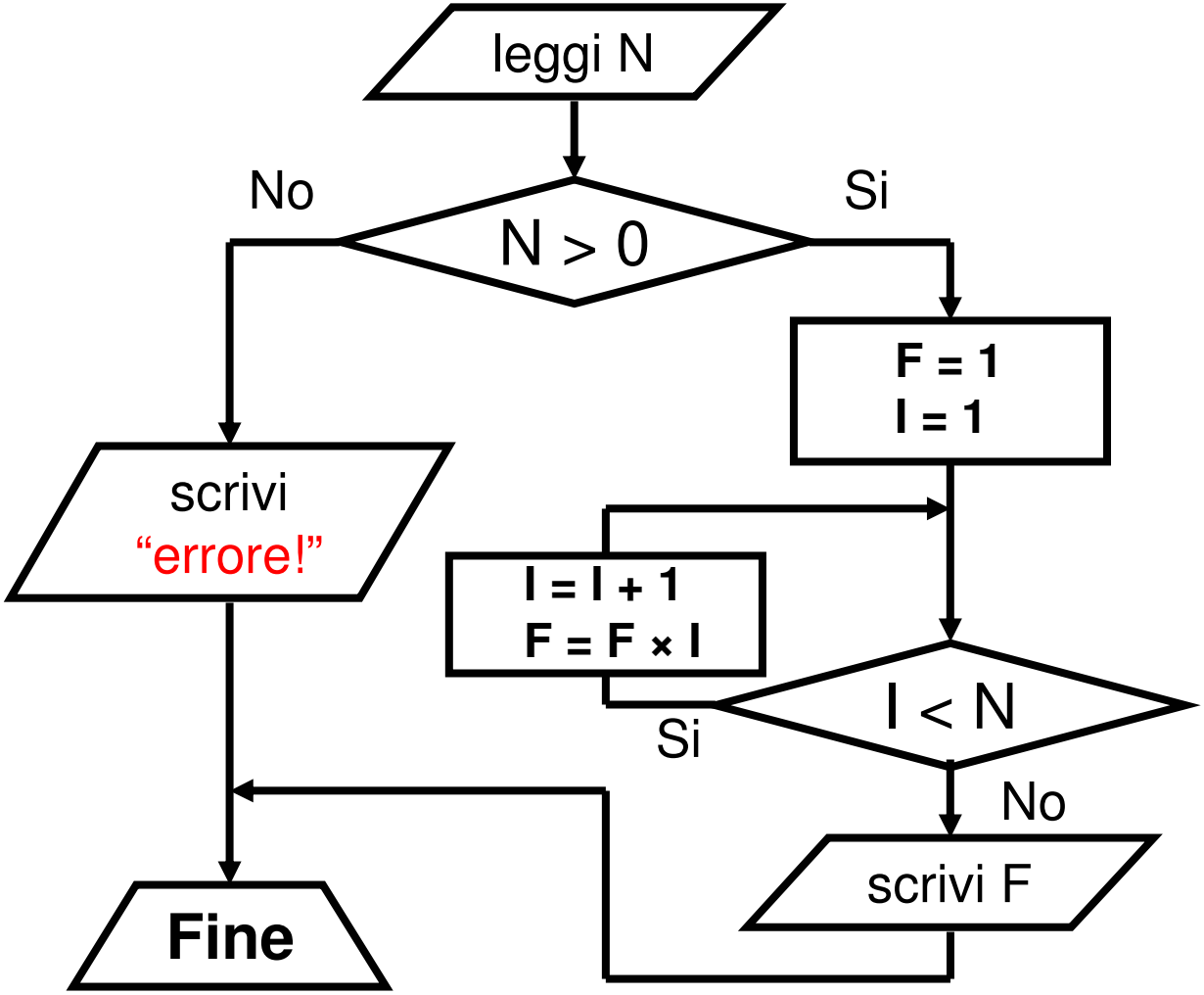
F		N		F
7	×	6	=	42
42	×	5	=	210
210	×	4	=	840
840	×	3	=	2520
2520	×	2	=	**5040**

Casi particolari:

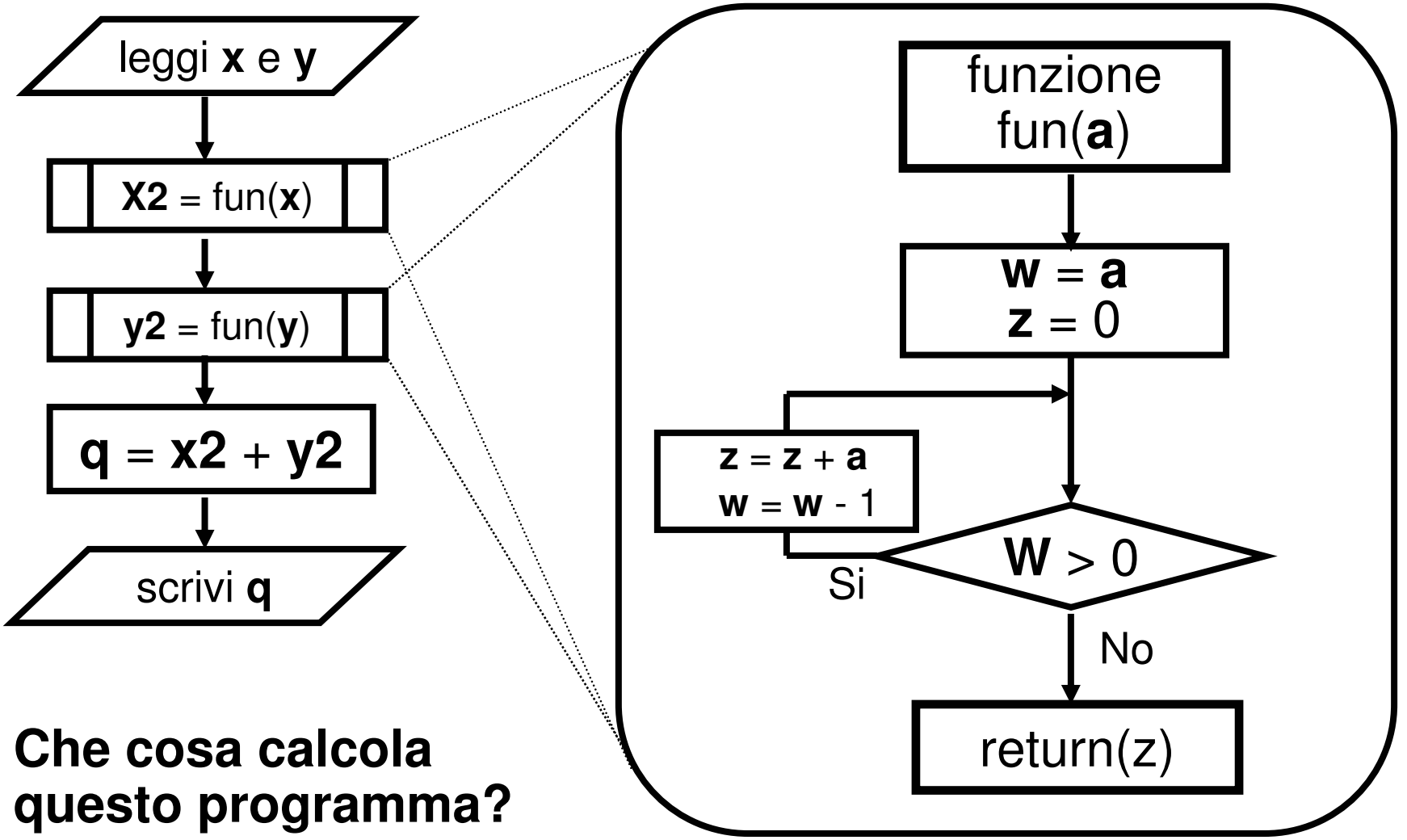
N = 0 produce F = 0

N = -4 Produce F = -4

Gestire le “eccezioni”



Sottoprogrammi



Che cosa calcola questo programma?

Esercizi

Esercizio 1

- L'esecutore deve leggere un numero N indicato da un utente esterno e deve poi calcolare e stampare la **somma di tutti i numeri compresi tra 0 e N**
- Si presti attenzione al fatto che il numero indicato può essere positivo, negativo o uguale a zero
- Ad esempio, se il numero indicato dall'utente esterno fosse 5, il risultato generato dall'esecutore dovrebbe essere **15** (i.e.0+1+2+3+4+5); se fosse invece -4, il risultato dovrebbe essere -10 (i.e. (-1)+(-2)+(-3)+(-4)).

Esercizio 2

- L'esecutore deve leggere una sequenza di numeri naturali (i.e. interi positivi maggiori di zero), calcolandone e stampandone il **minimo**
- La sequenza si interrompe non appena viene introdotto un numero negativo o nullo
- Ad esempio, data la sequenza 8, 3, 90, 2, -10, il risultato dovrebbe essere: "minimo = 2".

Esercizio 3

- L'esecutore deve leggere una sequenza di numeri naturali, calcolandone e stampandone il **massimo,** il **minimo** e la **media**
- La sequenza si interrompe non appena viene introdotto un numero negativo o nullo
- Ad esempio, data la sequenza 8, 3, 3, 2, -10, il risultato dovrebbe essere: "massimo = 8, minimo = 2, media = 4".

Esercizio 4

- L'esecutore deve leggere un numero N indicato da un utente esterno; questo numero indica la lunghezza della sequenza di numeri che verranno poi inseriti dallo stesso utente. Ad esempio, data la sequenza 3, 7, 10, -3, l'utente dovrà inserire: **4**, 3, 7, 10, -3
- Di questi numeri l'esecutore dovrà calcolare e stampare **massimo**, **minimo** e **media**, escludendo l'indicatore di lunghezza.

Esercizio 5

- L'esecutore deve leggere un numero intero che rappresenta l'anno, e stampare:
  - **"vero"** se l'anno in esame è **bisestile**
  - **"falso"** se l'anno **non** è **bisestile**
- Si ricordi che un anno è bisestile se:
  - è divisibile per 4 ma non per 100 oppure
  - è divisibile per 400
- Ad esempio, 1900 e 2100 non sono anni bisestili, mentre 1996, 2000 e 2004 lo sono.

Esercizio 6

- L'esecutore deve leggere **tre** numeri interi che rappresentano una data in termini di giorno, mese ed anno, e deve stampare il **numero di giorni trascorsi dall'inizio dell'anno**
- Si supponga per semplicità che i dati in ingresso siano sempre corretti.