LA PROGRAMMAZIONE STRUTTURATA

Introduzione

CC BY

INTRODUZIONE

- ❑ Analisi del problema
- ❑ Progettazione del software
- ❑ Implementazione del software
- ❑ Test (alfa e beta)
- ❑ Rilascio del software e della documentazione
- ❑ Manutenzione (help desk, fixing e upgrade)

CICLO DI VITA DEL SOFTWARE

Durante la fase di progettazione si deve stabilire – tra l'altro – quale **paradigma di programmazione** adoperare.

INTRODUZIONE

Un paradigma di programmazione concettualmente è un **insieme di strumenti** a disposizione del programmatore.

PARADIGMA DI PROGRAMMAZIONE

In definitiva la scelta è tra due soli paradigmi:

- la **programmazione strutturata** (storicamente più vecchio e più adatto a problemi semplici)
- la **programmazione object oriented** (più recente ed adatto alla progettazione di sistemi complessi che risolvono problemi altrettanto complessi)

INTRODUZIONE

In questo corso conosceremo gli strumenti della **programmazione strutturata**.

Nella programmazione strutturata la soluzione ad un problema è l'**algoritmo**.

Un **algoritmo** fa uso di due classi di strumenti:

- le **strutture dati**
- le **strutture di controllo del flusso**

Andiamo quindi a trattare i seguenti argomenti:

- algoritmo
- strutture dati
- strutture di controllo del flusso

LA PROGRAMMAZIONE STRUTTURATA

Il concetto e la definizione di algoritmo

CC BY

IL CONCETTO DI ALGORITMO

Sebbene la parola **algoritmo** suoni un po' difficile la nostra **vita quotidiana** è piena di algoritmi.

Chiedere informazioni.

PROBLEMA

1. **Prosegui diritto per 100 metri**
2. **Gira a destra**
3. **Alla fine della strada gira a destra**
4. **Al secondo semaforo gira a sinistra**
5. **Alla rotatoria prendi la terza uscita**
6. **Dopo 500 metri sei arrivato**

ALESSANDRO URSOMANDO

CC BY

IL CONCETTO DI ALGORITMO

Sebbene la parola **algoritmo** suoni un po' difficile la nostra **vita quotidiana** è piena di algoritmi.

Lavare il bucato con la lavatrice.

PROBLEMA

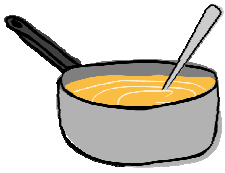
1. **Se l'oblò è chiuso, aprilo**
2. **Metti i panni nel cestello**
3. **Chiudi l'oblò**
4. **Accendi la lavatrice**
5. **Imposta il programma**
6. **Metti il detersivo**
7. **Metti l'ammorbidente**
8. **Avvia il ciclo**

CC BY

IL CONCETTO DI ALGORITMO

Sebbene la parola **algoritmo** suoni un po' difficile la nostra **vita quotidiana** è piena di algoritmi.

Preparare la crema pasticcera.



PROBLEMA

1. **Riscalda 500 ml di latte**
2. **Lavora 6 tuorli con 150 gr di zucchero e 50 gr di farina**
3. **Versa il composto nella casseruola**
4. **Continua a mescolare finché la crema assume la densità desiderata.**

CC BY

IL CONCETTO DI ALGORITMO.

Sebbene la parola **algoritmo** suoni un po' difficile la nostra **vita quotidiana** è piena di algoritmi.

Vedremo in seguito che **tutti** gli strumenti utili alla creazione di algoritmi sono importati dalla **vita quotidiana**.

CC BY

DEFINIZIONE DI ALGORITMO

Un algoritmo è una sequenza **ordinata** e **finita** di **passi elementari** che risolvono una **classe di problemi.**

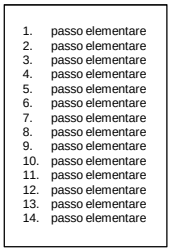
ALGORITMO

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare
12. passo elementare
13. passo elementare
14. passo elementare

DEFINIZIONE DI ALGORITMO

Un algoritmo è una sequenza **ordinata** e **finita** di **passi elementari** che risolvono una **classe di problemi.**

ALGORITMO

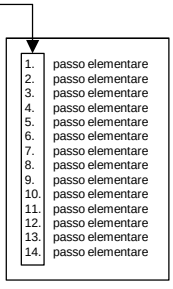


Un **passo elementare** non è ulteriormente scomponibile: è un comando chiaro e inequivocabile.

DEFINIZIONE DI ALGORITMO

Un algoritmo è una sequenza **ordinata** e **finita** di **passi elementari** che risolvono una **classe di problemi.**

ALGORITMO



Un **passo elementare** non è ulteriormente scomponibile: è un comando chiaro e inequivocabile.

DEFINIZIONE DI ALGORITMO

Un algoritmo è una sequenza **ordinata** e **finita** di **passi elementari** che risolvono una **classe di problemi.**

ALGORITMO

Un **passo elementare** non è ulteriormente scomponibile: è un comando chiaro e inequivocabile.

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare
12. passo elementare
13. passo elementare
14. passo elementare

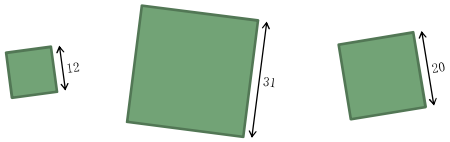
DEFINIZIONE DI ALGORITMO

Un algoritmo è una sequenza **ordinata** e **finita** di **passi elementari** che risolvono una **classe di problemi.**

ALGORITMO

Un **passo elementare** non è ulteriormente scomponibile: è un comando chiaro e inequivocabile.

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare
12. passo elementare
13. passo elementare
14. passo elementare



Un algoritmo raggiunge il suo obiettivo raccogliendo, manipolando e presentando delle **informazioni**.

LA PROGRAMMAZIONE STRUTTURATA

Strutture dati

CC BY

INFORMAZIONE E DATO

Un'**informazione** archiviata in un computer prende il nome di **dato**.

Il **dato** viene tenuto in una **struttura dati** della quale conosciamo il **tipo**, il **nome** ed il suo **indirizzo** in memoria centrale o in memoria di massa

DATO

Classifichiamo i dati in:
- dati di **input**
- dati di **output**
- dati **interni**

ALESSANDRO URSOMANDO

CC BY

INFORMAZIONE E DATO

Il **dato** viene tenuto in una **struttura dati** della quale conosciamo il **tipo**, il **nome** ed il suo **indirizzo** in memoria centrale o di memoria di massa

DATO

- ❑ variabili
- ❑ costanti

STRUTTURE DATI SEMPLICI

- ❑ vettori
- ❑ record

STRUTTURE DATI COMPLESSE

- ❑ file

S.D. SU MEMORIA DI MASSA

- ❑ puntatori

STRUTTURE DATI DINAMICHE

CC BY

VARIABILI..

13

Una variabile è un'area della RAM riservata per un uso specifico. Ci si riferisce ad essa mediante un **nome**; essa ha *sempre* un **valore**; i valori ammessi sono tutti dello stesso **tipo**.

VARIABILE

Dichiarare una variabile vuol dire riservare lo spazio in memoria.
Ogni variabile viene dichiarata una volta sola.

L'**assegnazione** è la procedura mediante la quale si definisce il valore della variabile. Il valore di una variabile di solito cambia diverse volte.

CC BY

..E COSTANTI

Una variabile è un'area della RAM riservata per un uso specifico. Ci si riferisce ad essa mediante un **nome**; essa ha *sempre* un **valore**; i valori ammessi sono tutti dello stesso **tipo**.

VARIABILE

Una costante è un'area della RAM con un **nome**, un **valore** ed un **tipo**. Differisce dalla variabile perché una volta assegnatole un valore questo non cambia *mai più.*

COSTANTE

CC BY

VETTORI

Un **vettore** è un contenitore per una certa quantità di dati **omogenei** (dello stesso tipo).

Il vettore ha un **nome**, un **tipo** (valido per ciascun elemento del vettore) e una **dimensione** (che indica la quantità massima di elementi).

Ogni cella del vettore è individuata univocamente con un numero – detto **indice** – che va da 0 a n-1 (dove n è la dimensione del vettore).

VETTORE

Per esempio, se **libri** è il nome del vettore che contiene i titoli di tutti i vostri libri di testo con **libri[0]** indichiamo il primo libro.

CC BY

RECORD

Un **record** è un contenitore per una certa quantità di dati **non omogenei** (di tipo diverso).

Il record è caratterizzato da un **nome proprio**, ed un elenco di elementi per i quali si indicano un **nome** ed un **tipo**.

RECORD

Per esempio, se **persona** è il nome del record possiamo immaginare che esso contenga gli elementi **nome** (di tipo stringa), **cognome** (di tipo stringa), **altezzaCm** (di tipo numerico-intero), **pesoKg** (di tipo numerico-reale) e **maggiorenne** (di tipo booleano).

ALESSANDRO URSOMANDO

CC BY

FILE

I file sono l'unico modo per archiviare informazioni in modo definitivo (quindi su memoria di massa e non in RAM).

I linguaggi di programmazione di alto livello forniscono il modo di interagire con i file (che sono gestiti dal sistema operativo): sarà quindi possibile creare, aprire in lettura, aprire in lettura/scrittura e chiudere i file.

FILE

A livello teorico esistono due tipi di file: i **file di testo** ed i file di tipo strutturato (o **file di record**).

ALESSANDRO URSOMANDO

CC BY

PUNTATORI

Un puntatore è una variabile il cui contenuto è l'indirizzo di un'altra variabile.

PUNTATORE

INDIRIZZO

Scaffale 11 – Posto 8

LA PROGRAMMAZIONE STRUTTURATA

Strutture dati – I puntatori

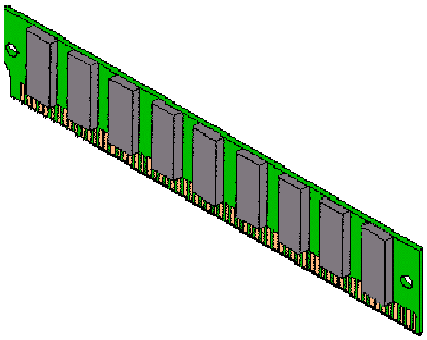
[corso avanzato]

CC BY

PUNTATORI: PROLOGO

La RAM è il posto dove vengono memorizzate tutte le informazioni relative ai processi :

- il codice
- i dati
- tutte le strutture dati necessarie a gestirne l'esecuzione



Ogni informazione ha la sua locazione di memoria.

Quante sono queste locazioni di memoria?

CC BY

PUNTATORI: PROLOGO

La RAM è il posto dove vengono memorizzate tutte le informazioni relative ai processi :

- il codice
- i dati
- tutte le strutture dati necessarie a gestirne l'esecuzione

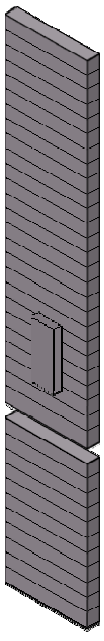
Ogni informazione ha la sua locazione di memoria.

Quante sono queste locazioni di memoria?

In un'architettura a 32 bit possiamo indirizzare 2^{32} locazioni di memoria.

Se utilizziamo la notazione esadecimale avremo che la prima locazione avrà indirizzo $0000\ 0000_{16}$ e l'ultima $FFFF\ FFFF_{16}$.

Questo schema di indirizzamento è una nostra semplificazione: in realtà i sistemi operativi hanno gestori della memoria molto complessi.



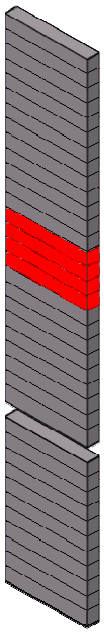
CC BY

PUNTATORI

Quando dichiariamo una variabile il sistema provvede ad assegnarle la quantità di memoria necessaria ed a prendere nota (in una apposita struttura di sistema) del nome scelto, del tipo (il che esprime la quantità di RAM che usa) e l'indirizzo della prima locazione di memoria riservata ad essa.

Nome Variabile	Tipo	Indirizzo
x	intero	1000 1000

Supponiamo di dichiarare una certa variabile **x** di tipo intero. Visto che per un intero servono 4 byte, il sistema trova 4 locazioni consecutive libere e le assegna ad **x**, tenendo traccia di tutto ciò nella struttura di sistema preposta.



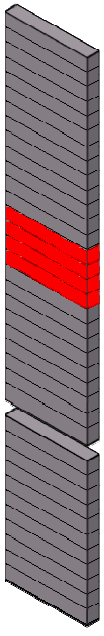
CC BY

PUNTATORI

Successivamente, quando il sistema dovrà leggere il contenuto di **x**, consultando la struttura dati preposta, saprà quante e quali locazioni leggere e come decodificare i bit in esse presenti.

Analogamente per la scrittura di **x**.

Nome Variabile	Tipo	Indirizzo
x	intero	1000 1000



CC BY

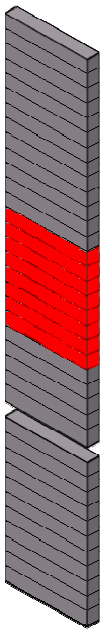
PUNTATORI

Successivamente, quando il sistema dovrà leggere il contenuto di **x**, consultando la struttura dati preposta, saprà quante e quali locazioni leggere e come decodificare i bit in esse presenti.

Analogamente per la scrittura di **x**.

Supponiamo di dichiarare una certa variabile **y** di tipo puntatore. Visto che anche per un puntatore servono 4 byte, il sistema trova 4 locazioni consecutive libere e le assegna ad **y**.

Nome Variabile	Tipo	Indirizzo
x	intero	1000 1000
y	puntatore a intero	1000 1004



ALESSANDRO URSOMANDO

CC BY

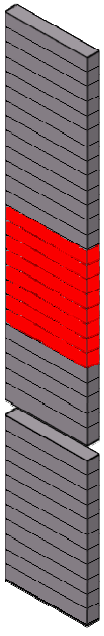
PUNTATORI

A questo punto così come posso inserire un valore numerico in **x**, posso inserire un indirizzo in **y**.

Supponiamo quindi di inserire l'indirizzo di **x** (1000 1000) in **y**.

Successivamente potrò accedere alla struttura dati preposta, vedere dove è situata la variabile **y** (1000 1004), accedere al suo valore (1000 1000) ed usare lo stesso per accedere nuovamente alla RAM. Questo nuovo accesso alla RAM mi riporta al valore di **x**.

Nome Variabile	Tipo	Indirizzo
x	intero	1000 1000
y	puntatore a intero	1000 1004



ALESSANDRO URSOMANDO

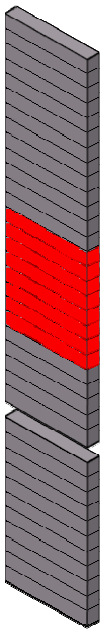
CC BY

PUNTATORI

Le potenzialità dei puntatori sono molteplici ma esulano dagli obiettivi di questo corso.

L'unico uso che faremo noi dei puntatori sarà per il passaggio di parametri per indirizzo, più avanti in questo corso.

Nome Variabile	Tipo	Indirizzo
x	intero	1000 1000
y	puntatore a intero	1000 1004



ALESSANDRO URSOMANDO

LA PROGRAMMAZIONE STRUTTURATA

Strutture di controllo del flusso: introduzione

CC BY

RACCOGLIAMO LE IDEE

- Ciclo di vita del software
  - Fase di progettazione
- Paradigma di programmazione
  - Programmazione strutturata
- Algoritmo
  - Strutture dati
  - Strutture di controllo del flusso

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare
12. passo elementare
13. passo elementare
14. passo elementare

Sono rimasti in sospeso i concetti di:

- Struttura di controllo del flusso
- Passo elementare

CC BY

PASSI ELEMENTARI E STRUTTURE DI CONTROLLO DEL FLUSSO

Un passo elementare può servire a:

- Acquisire un input
- Presentare un output
- Modificare il valore di una struttura dati
- Dirigere il flusso di esecuzione

Qualsiasi algoritmo, anche quelli del mondo reale, per adattarsi alle circostanze potrebbe non richiedere l'esecuzione in sequenza di tutti i suoi passi elementari.

In altre parole, non è detto che dopo il passo 6 si esegua **sempre** il passo 7.

Vediamo qualche esempio.

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare
12. passo elementare
13. passo elementare
14. passo elementare

CC BY

PASSI ELEMENTARI E STRUTTURE DI CONTROLLO DEL FLUSSO

Chiedere informazioni.

PROBLEMA

In un algoritmo come questo, i passi da eseguire sono banali.

1. **Prosegui diritto per 100 metri**
2. **Gira a destra**
3. **Alla fine della strada gira a destra**
4. **Al secondo semaforo gira a sinistra**
5. **Alla rotatoria prendi la terza uscita**
6. **Dopo 500 metri sei arrivato**

CC BY

PASSI ELEMENTARI E STRUTTURE DI CONTROLLO DEL FLUSSO

Lavare il bucato con la lavatrice.

PROBLEMA

1. **Se l'oblò è chiuso, aprilo**
2. **Metti i panni nel cestello**
3. **Chiudi l'oblò**
4. **Accendi la lavatrice**
5. **Imposta il programma**
6. **Metti il detersivo**
7. **Metti l'ammorbidente**
8. **Avvia il ciclo**

CC BY

PASSI ELEMENTARI E STRUTTURE DI CONTROLLO DEL FLUSSO

Lavare il bucato con la lavatrice.

PROBLEMA

Qui, il passo uno è scomponibile in due passi, il secondo dei quali potrebbe non essere eseguito tutte le volte.

1. Se l'oblò è chiuso, aprilo
2. Metti i panni nel cestello
3. Chiudi l'oblò
4. Accendi la lavatrice
5. Imposta il programma
6. Metti il detersivo
7. Metti l'ammorbidente
8. Avvia il ciclo

1. Se l'oblò è chiuso,
   2. Apri l'oblò
3. Metti i panni nel cestello
4. Chiudi l'oblò
5. Accendi la lavatrice
6. Imposta il programma
7. Metti il detersivo
8. Metti l'ammorbidente
9. Avvia il ciclo

ALESSANDRO URSOMANDO

CC BY

PASSI ELEMENTARI E STRUTTURE DI CONTROLLO DEL FLUSSO

Preparare la crema pasticcera.

PROBLEMA

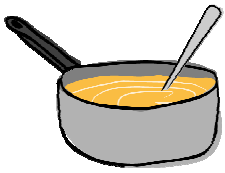
1. Riscalda 500 ml di latte
2. Lavora 6 tuorli con 150 gr di zucchero e 50 gr di farina
3. Versa il composto nella casseruola
4. Continua a mescolare finché la crema assume la densità desiderata.

ALESSANDRO URSOMANDO

CC BY

PASSI ELEMENTARI E STRUTTURE DI CONTROLLO DEL FLUSSO

Preparare la crema pasticcera.



PROBLEMA

1. Riscalda 500 ml di latte
2. Lavora 6 tuorli con 150 gr di zucchero e 50 gr di farina
3. Versa il composto nella casseruola
4. Continua a mescolare finché la crema assume la densità desiderata.

Qui, il passo quattro prevede la ripetizione di una certa attività.

1. Riscalda 500 ml di latte
2. Lavora 6 tuorli con 150 gr di zucchero e 50 gr di farina
3. Versa il composto nella casseruola
4. Mescola
5. Se la densità non ti soddisfa torna al punto 4.

ALESSANDRO URSOMANDO

CC BY

PASSI ELEMENTARI E STRUTTURE DI CONTROLLO DEL FLUSSO

Le strutture di controllo del flusso sono tre.

I passi vengono eseguiti uno dopo l'altro incondizionatamente

SEQUENZA

I passi vengono eseguiti solo dopo avere verificato una condizione

SELEZIONE

I passi elementari vengono eseguiti più volte

ITERAZIONE

Vediamole nel dettaglio nelle sezioni che seguono.

ALESSANDRO URSOMANDO

LA PROGRAMMAZIONE STRUTTURATA

Strutture di controllo del flusso: sequenza

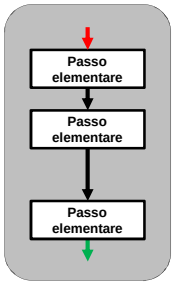
CC BY

SEQUENZA

La **sequenza** è la struttura di controllo più banale, vediamo qualche flow chart di esempio.

I passi vengono eseguiti uno dopo l'altro incondizionatamente

SEQUENZA



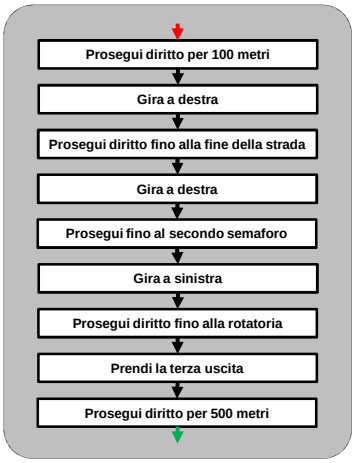
ALESSANDRO URSOMANDO

CC BY

SEQUENZA

Chiedere informazioni.

PROBLEMA DEL MONDO REALE



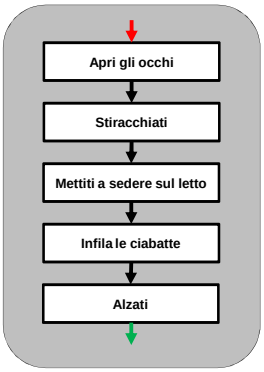
ALESSANDRO URSOMANDO

CC BY

SEQUENZA

Svegliarsi.

PROBLEMA DEL MONDO REALE



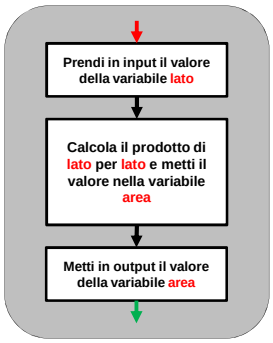
ALESSANDRO URSOMANDO

CC BY

SEQUENZA

Calcolare l'area di un quadrato.

PROBLEMA DI INFORMATICA



LA PROGRAMMAZIONE STRUTTURATA

Strutture di controllo del flusso: selezione

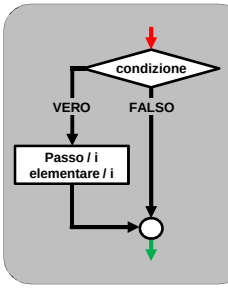
CC BY

SELEZIONE

La selezione può essere di 3 tipi.
Cominciamo dal più banale: **selezione a una via**.

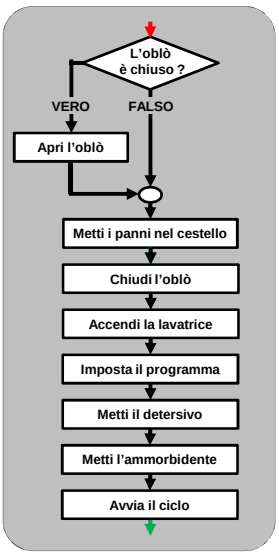
I passi vengono eseguiti solo dopo avere verificato una condizione

SELEZIONE



CC BY

SELEZIONE A UNA VIA

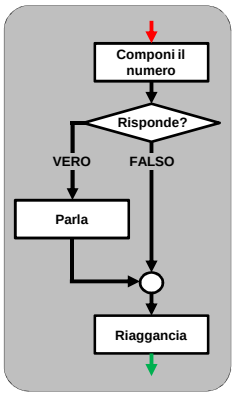


Lavare il bucato con la lavatrice.

PROBLEMA DEL MONDO REALE

CC BY

SELEZIONE A UNA VIA

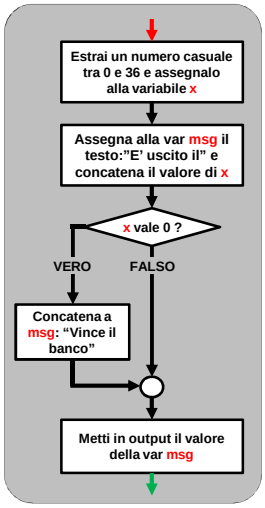


Fare una telefonata.

PROBLEMA DEL MONDO REALE

CC BY

SELEZIONE A UNA VIA



Simulare una sessione di gioco alla roulette.

PROBLEMA DI INFORMATICA

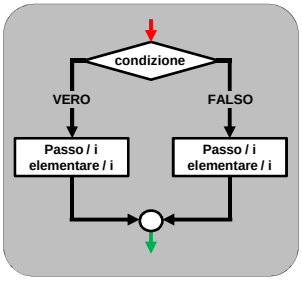
CC BY

SELEZIONE

La **selezione a due vie** prevede delle operazioni da svolgere sia nel caso in cui la condizione risulti vera, sia nel caso risulti falsa.

I passi vengono eseguiti solo dopo avere verificato una condizione

SELEZIONE

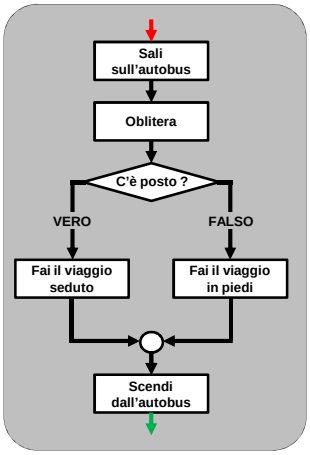


CC BY

SELEZIONE A DUE VIE

Fare un viaggio in autobus.

PROBLEMA DEL MONDO REALE

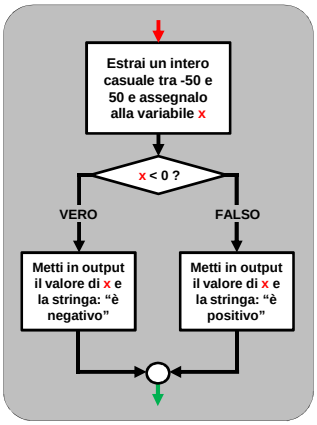


CC BY

SELEZIONE A DUE VIE

Riconoscere il segno di un numero casuale.

PROBLEMA DI INFORMATICA



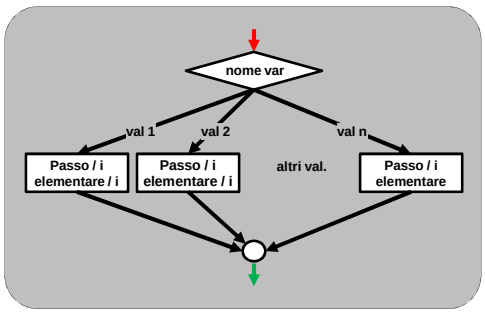
CC BY

SELEZIONE

La **selezione ad n vie** prevede delle operazioni da svolgere in diversi casi.

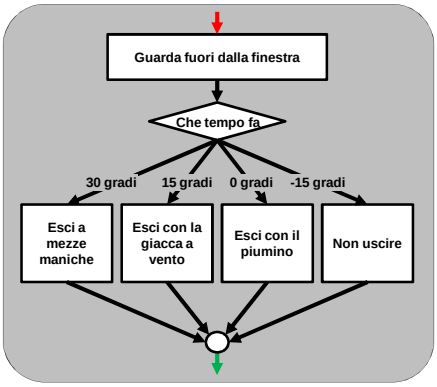
I passi vengono eseguiti solo dopo avere verificato una condizione

SELEZIONE



CC BY

SELEZIONE AD N VIE

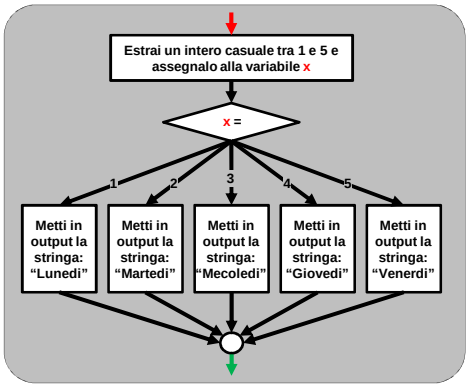


Decidere cosa indossare.

PROBLEMA DEL MONDO REALE

CC BY

SELEZIONE AD N VIE



Estrarre un giorno lavorativo a caso.

PROBLEMA DI INFORMATICA

LA PROGRAMMAZIONE STRUTTURATA

Strutture di controllo del flusso: iterazione

CC BY

ITERAZIONE

Sebbene il concetto di iterazione sia semplice qui la faccenda si complica perché esistono 3 tipi di cicli.

I passi elementari vengono eseguiti più volte

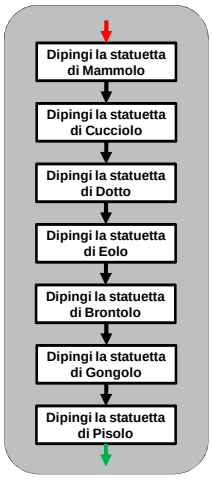
ITERAZIONE

La difficoltà sta nel capire in che circostanze è più opportuno usare l'uno o l'altro.

Cominciamo, come al solito, dal più semplice.

CC BY

ITERAZIONE



Il concetto di ciclo è stato introdotto per alleggerire situazioni come questa.

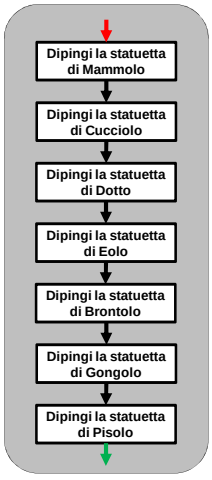
Dipingere le statuette dei nani

PROBLEMA DEL MONDO REALE

ALESSANDRO URSOMANDO

CC BY

ITERAZIONE



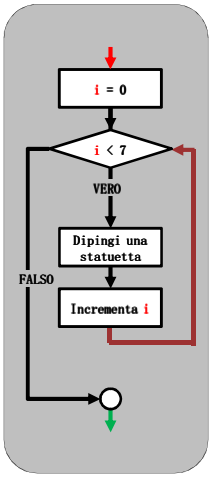
Il concetto di ciclo è stato introdotto per alleggerire situazioni come questa.

Una volta isolata l'istruzione (o le istruzioni) da eseguire **più volte** potremmo scrivere un algoritmo che semplicemente esegua quelle istruzioni (o quella istruzione) quante volte sono necessarie.

ALESSANDRO URSOMANDO

CC BY

ITERAZIONE



Il concetto di ciclo è stato introdotto per alleggerire situazioni come questa.

Una volta isolata l'istruzione (o le istruzioni) da eseguire **più volte** potremmo scrivere un algoritmo che semplicemente esegua quelle istruzioni (o quella istruzione) quante volte sono necessarie.

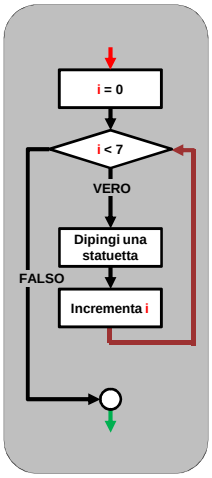
Per far ciò useremo una **variabile** con la quale contiamo quante volte abbiamo eseguito il ciclo.

Questo tipo di variabile viene denominata **contatore** e di solito il suo nome è **i**.

ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONTATORE



Il concetto di ciclo è stato introdotto per alleggerire situazioni come questa.

Una volta isolata l'istruzione (o le istruzioni) da eseguire **più volte** potremmo scrivere un algoritmo che semplicemente esegua quelle istruzioni (o quella istruzione) quante volte sono necessarie.

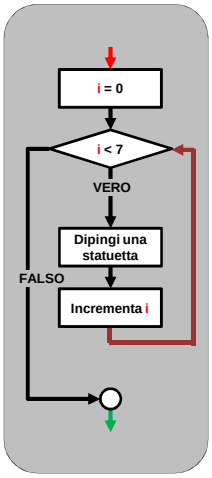
Per far ciò useremo una **variabile** con la quale contiamo quante volte abbiamo eseguito il ciclo.

Questo tipo di variabile viene denominata **contatore** e di solito il suo nome è **i**.

ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONTATORE

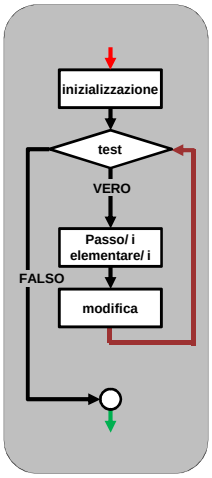


La variabile contatore viene quindi **inizializzata** prima di fare il ciclo, **testata** prima di ripetere il ciclo e **modificata** una volta eseguito il ciclo.

ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONTATORE

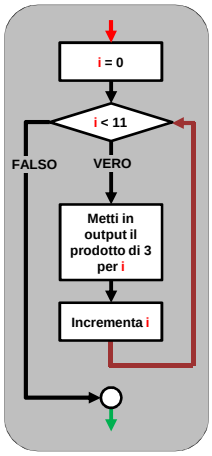


La variabile contatore viene quindi **inizializzata** prima di fare il ciclo, **testata** prima di ripetere il ciclo e **modificata** una volta eseguito il ciclo.

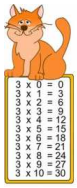
ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONTATORE



Stampare la tabellina del 3.



PROBLEMA DI INFORMATICA

ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE

Se non sappiamo a priori **quante** volte dobbiamo eseguire il ciclo, sicuramente sappiamo **quando** eseguirlo e **quando** no.

I passi elementari vengono eseguiti più volte

ITERAZIONE

Anche questa volta ci sono due casi.

Prima eseguiamo il ciclo e poi facciamo il test? Oppure prima facciamo il test e poi eseguiamo il ciclo ?

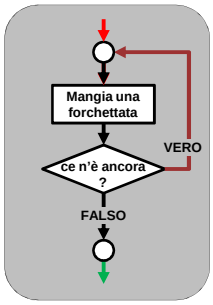
ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE

Mangiare un piatto di spaghetti.

PROBLEMA DEL MONDO REALE

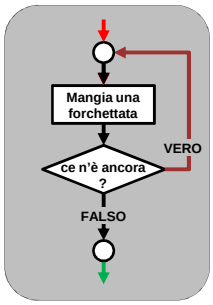


CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE CON CONDIZIONE IN CODA

Mangiare un piatto di spaghetti.

PROBLEMA DEL MONDO REALE

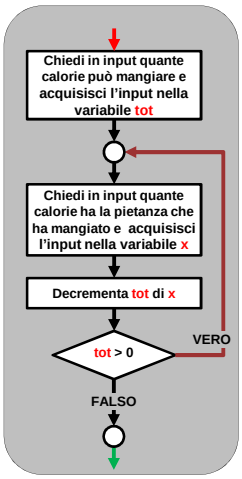


CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE CON CONDIZIONE IN CODA

Tenere sotto controllo le calorie.

PROBLEMA DI INFORMATICA

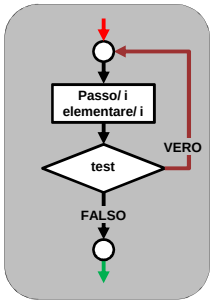


CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE CON CONDIZIONE IN CODA

Il ciclo su condizione con verifica della condizione in coda si presta in due casi:

- Quando ha senso porsi la domanda **solo dopo** avere già eseguito una prima volta il ciclo (come abbiamo appena visto).
- Quando c'è un obiettivo da raggiungere mediante uno o più **tentativi** (vediamo un esempio).

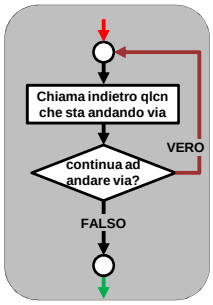


CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE CON CONDIZIONE IN CODA

Chiamare indietro qualcuno.

PROBLEMA DEL MONDO REALE

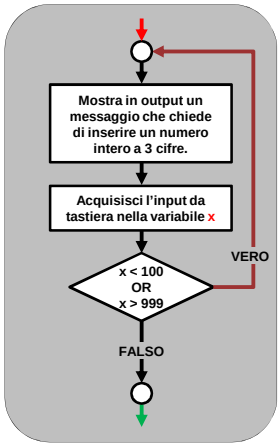


CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE CON CONDIZIONE IN CODA

Input controllato

PROBLEMA DI INFORMATICA



CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE

Quando ha senso porsi la domanda **prima** di eseguire la prima volta il ciclo..

I passi elementari vengono eseguiti più volte

ITERAZIONE

..allora si pone la verifica della condizione in testa.

Questo tipo di ciclo si chiama **ciclo di ripetizione su condizione con verifica della condizione in testa**.

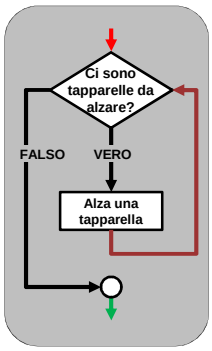
ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE CON CONDIZIONE IN TESTA

Alzare tutte le tapparelle di casa
(considerando che potrebbero essere già tutte aperte)

PROBLEMA DEL MONDO REALE



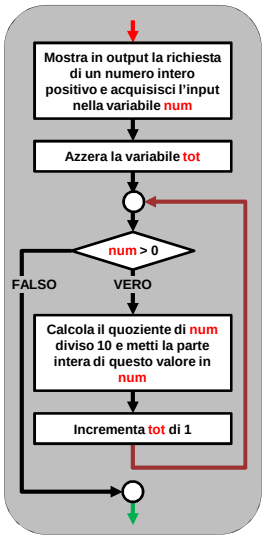
ALESSANDRO URSOMANDO

CC BY

CICLO DI RIPETIZIONE SU CONDIZIONE CON CONDIZIONE IN TESTA

Contare le cifre di un numero

PROBLEMA DI INFORMATICA

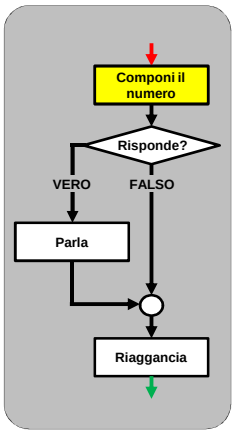


LA PROGRAMMAZIONE STRUTTURATA

Sviluppo top-down

CC BY

COMBINAZIONE DI STRUTTURE PER IL CONTROLLO DEL FLUSSO



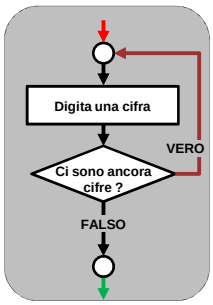
Fare una telefonata.

PROBLEMA DEL MONDO REALE

In realtà, il passo "Componi il numero" non è esattamente un passo elementare in quanto è scomponibile.

CC BY

COMBINAZIONE DI STRUTTURE PER IL CONTROLLO DEL FLUSSO



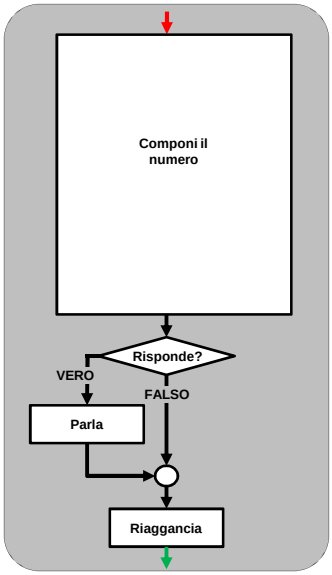
Comporre un numero.

PROBLEMA DEL MONDO REALE

In questo modo l'esercizio precedente diventa..

CC BY

COMBINAZIONE DI STRUTTURE PER IL CONTROLLO DEL FLUSSO

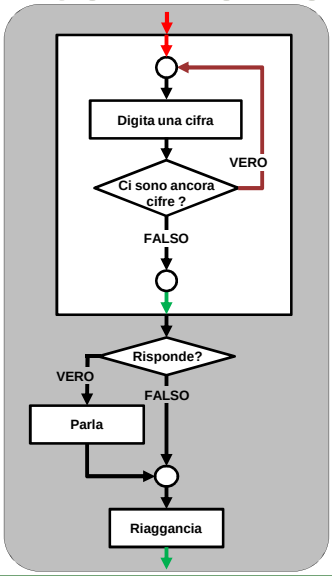


Fare una telefonata.

PROBLEMA DEL MONDO REALE

CC BY

COMBINAZIONE DI STRUTTURE PER IL CONTROLLO DEL FLUSSO



Fare una telefonata.

PROBLEMA DEL MONDO REALE

Le strutture di controllo possono essere combinate tra loro: questo principio è alla base dell'approccio **top-down**.

SVILUPPO TOP-DOWN

Per sviluppo **top-down** si intende un approccio alla soluzione di un problema che prevede la scomposizione dello stesso in sotto-problemi (più piccoli e quindi più semplici da risolvere).

SVILUPPO TOP-DOWN

Le strutture di controllo possono essere combinate tra loro: questo principio è alla base dell'approccio **top-down**.

SVILUPPO TOP-DOWN

Per sviluppo **top-down** si intende un approccio alla soluzione di un problema che prevede la scomposizione dello stesso in sotto-problemi (più piccoli e quindi più semplici da risolvere).

SVILUPPO TOP-DOWN

Evidentemente ogni **sotto-problema** sarà risolto con la realizzazione di un **sotto-programma**.

Evidentemente uno dei passi elementari del nostro algoritmo potrebbe essere la richiesta di esecuzione di uno dei **sotto-programmi**.

CC BY

SVILUPPO TOP-DOWN

Lo svolgimento di un compito ben preciso (sebbene piccolo) può essere svolto in autonomia, ma più spesso necessita di scambiare dei dati con chi si serve del sotto-programma.

Per esempio, la composizione di un numero richiede che si fornisca il numero da comporre!

I dati che un sotto-programma riceve in input da chi lo invoca si chiamano **parametri**.

Il dato che un sotto-programma fornisce in output può essere chiamato **valore restituito**.

CC BY

ESEMPIO

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare
12. passo elementare

1859. passo elementare
1860. passo elementare
1861. passo elementare
1862. passo elementare
1863. passo elementare
1864. passo elementare
1865. passo elementare

Supponiamo di avere un programma (che questo sia il programma principale o un sotto-programma non ha alcuna importanza)

Supponiamo poi di avere un sotto-programma qualsiasi.

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare

127. passo elementare
128. passo elementare
129. passo elementare
130. passo elementare
131. passo elementare

ESEMPIO

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare
12. passo elementare

1859. passo elementare
1860. passo elementare
1861. passo elementare
1862. passo elementare
1863. passo elementare
1864. passo elementare
1865. passo elementare

Quando progettiamo un sotto-programma dobbiamo – tra l'altro – indicare i parametri di cui necessita (**parametri formali**).

Supponiamo che esso si riferisca a questi parametri con i nomi a, b ,c, d.

a
b
c
d

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare

127. passo elementare
128. passo elementare
129. passo elementare
130. passo elementare
131. passo elementare

ESEMPIO

1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare
6. passo elementare
7. passo elementare
8. passo elementare
9. passo elementare
10. passo elementare
11. passo elementare

139. Invoca il sotto-programma qui a lato con i seguenti parametri: n1, n2, n3, n4 e metti il valore restituito nella variabile x.

1859. passo elementare
1860. passo elementare
1861. passo elementare
1862. passo elementare
1863. passo elementare
1864. passo elementare
1865. passo elementare

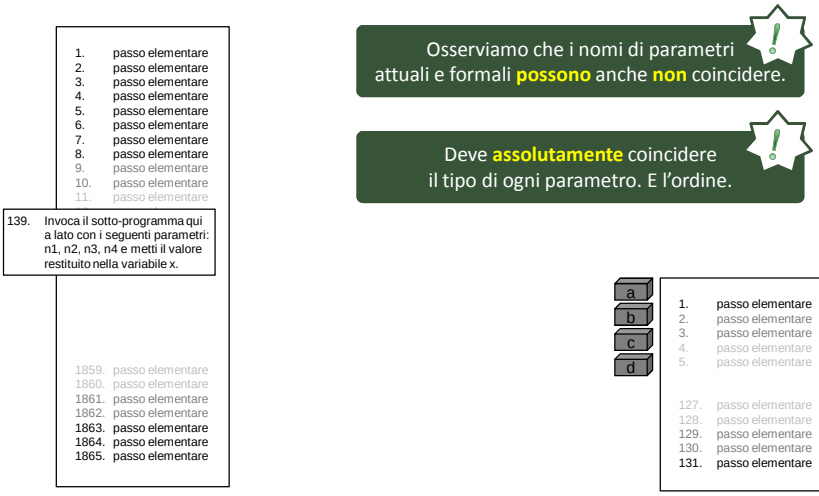
L'invocazione (da parte del programma chiamante) avviene indicando il nome del sotto-programma, fornendo i parametri (**parametri attuali**) con cui lavorare ed indicando cosa fare del valore restituito.

a
b
c
d

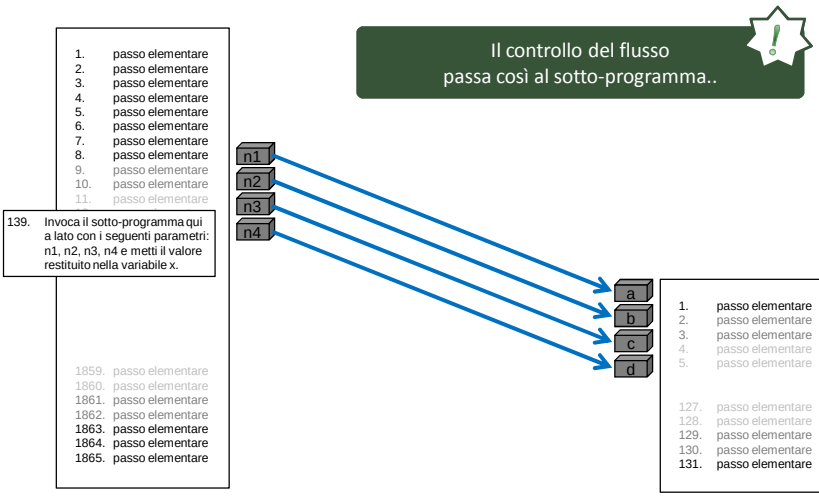
1. passo elementare
2. passo elementare
3. passo elementare
4. passo elementare
5. passo elementare

127. passo elementare
128. passo elementare
129. passo elementare
130. passo elementare
131. passo elementare

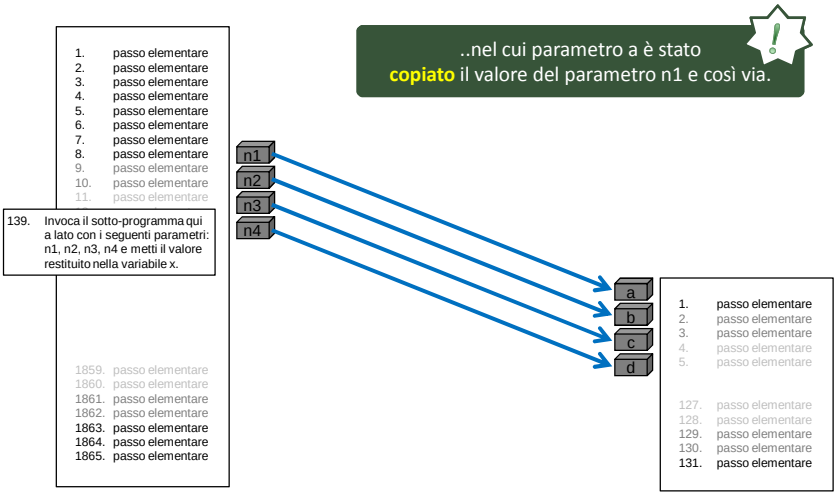
ESEMPIO



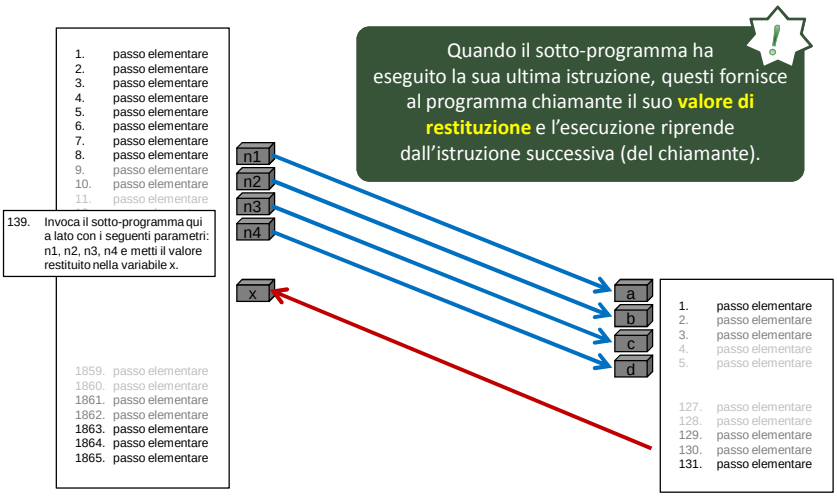
ESEMPIO



ESEMPIO



ESEMPIO



CC BY

SVILUPPO TOP-DOWN

Nello scenario che abbiamo appena descritto, possiamo avere qualsiasi numero di parametri ma sempre **uno ed un solo** valore restituito!

Se il sotto-programma necessita di restituire **altre informazioni** oltre al cosiddetto valore restituito, dobbiamo lavorare sui parametri.

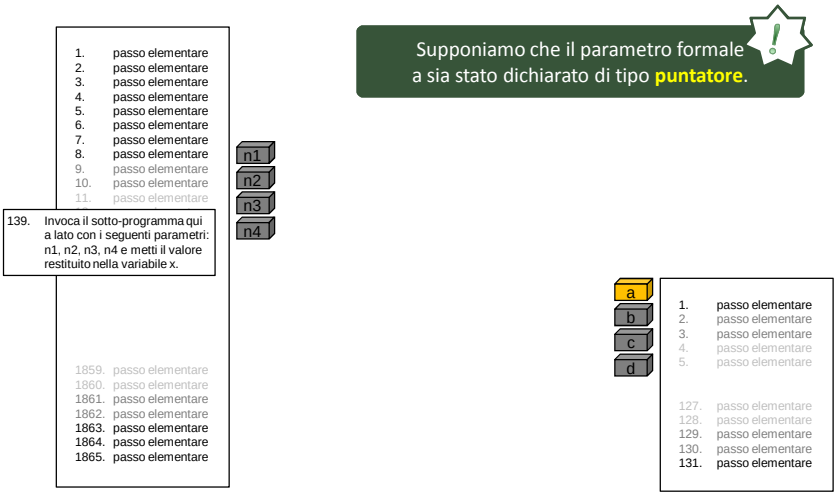
Nello scenario che abbiamo appena descritto il passaggio di parametri avviene **per copia**. Ovvero i valori dei parametri attuali vengono copiati nei parametri formali.

Ma se – nel progettare il sotto-programma – usiamo delle **variabili di tipo puntatore** come parametri (invece di usare delle variabili semplici), allora il passaggio di parametri avviene **per indirizzo**.

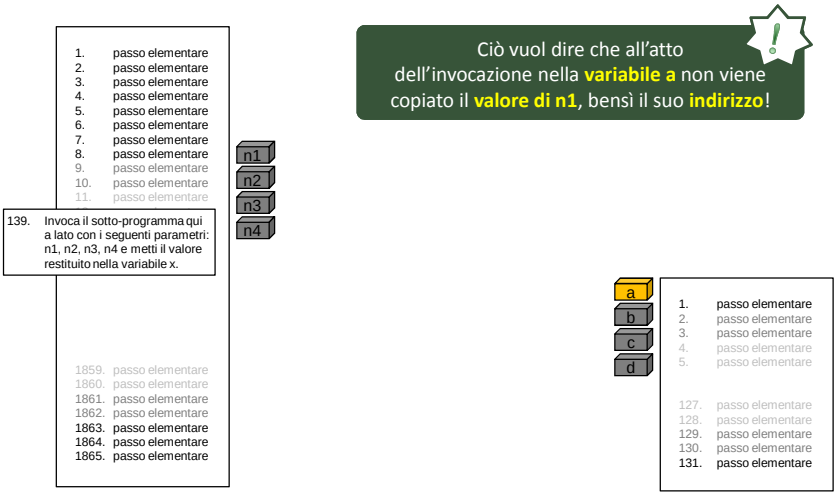
CC BY

ESEMPIO

Supponiamo che il parametro formale a sia stato dichiarato di tipo **puntatore**.

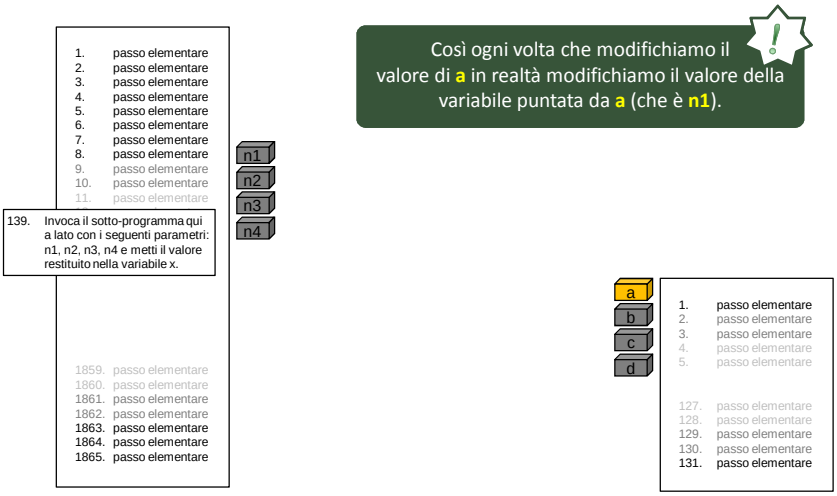


ESEMPIO



Ciò vuol dire che all'atto dell'invocazione nella **variabile a** non viene copiato il **valore di n1**, bensì il suo **indirizzo**!

ESEMPIO



Così ogni volta che modifichiamo il valore di **a** in realtà modifichiamo il valore della variabile puntata da **a** (che è **n1**).

LA PROGRAMMAZIONE STRUTTURATA

Istruzioni

CC BY

ISTRUZIONI

Riassumendo,
ogni passo elementare
sarà una istruzione del seguente elenco.

- di input
- di output
- di assegnazione
- di selezione
- di iterazione
- di invocazione di funzione

ISTRUZIONI

ISTRUZIONI DI SELEZIONE

ISTRUZIONI DI ITERAZIONE

ALESSANDRO URSOMANDO