Introduzione al Corso

Informatica B, AA 2017/2018

Giacomo Boracchi

giacomo.boracchi@polimi.it

Per Evitare Confusione

Prof. Masseroli ha lo scaglione ??? - PAD

Prof. Boracchi ha lo scaglione PAD - SAL

Prof. Cassano ha lo scaglione SAL - ZZZ

Prof. Cassano si trova ora in Aula B6.28

Prof. Masseroli inizia Giovedì mattina

La suddivisione degli studenti nelle varie sezioni potrebbe cambiare nei prossimi giorni (riguarda solamente gli studenti "vicini" a PAD o a SAL).

Chi siamo

Chi siamo

Giacomo Boracchi (docente)

- Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano
- homepage: http://home.deib.polimi.it/boracchi/
- email giacomo.boracchi@polimi.it
- ricevimento studenti:
  - Lunedì 9.30 – 11.30 nel mio Ufficio
  - **Su appuntamento**

  In **ogni caso** è consigliabile **prendere appuntamento via email.**
- Ufficio 157 via Ponzio 34/5, Milano.
- tel 02 2399 3467

Chi siamo

Ing. Francesco Trovò (esercitatore)

- homepage: http://home.deib.polimi.it/trovo/
- email: francesco1.trovo@polimi.it
- Pagina esercitazioni
  http://home.deib.polimi.it/trovo/infob_2017_2018.html

Chi siamo

Marco Santambrogio (responsabile di laboratorio)

- homepage: http://home.dei.polimi.it/santambr/
- email: marco.santambrogio@polimi.it
- Pagina Laboratorio

http://home.deib.polimi.it/santambr/dida/infob/1718/labgb/

Dr. Diego Carrera (responsabile di laboratorio)

- homepage: https://home.deib.polimi.it/carrerad/
- email diego.carrera@polimi.it

Il Corso

http://home.deib.polimi.it/boracchi/teaching/InfoB.htm

Organizzazione

Lezioni: 36 ore

il Mercoledì dalle 8.45 alle 11.15 in aula L.08

Esercitazioni: 28 ore

il Venerdì dalle 12.15 alle 14.15 in aula B2.2.1 (ex CT.60)

Laboratori: 18 ore (6 incontri)

Il Lunedì dalle 15.15 alle 18.15 in B.6.27

A volte **recupereremo lezioni** o **esercitazioni il Lunedì** dalle 15.15 in B.6.27

Controllate sempre il **calendario** del corso:

http://home.deib.polimi.it/boracchi/teaching/InfoBCalendar.htm

Argomenti Trattati

- Introduzione all'informatica
- Codifica binaria e algebra di Boole
- Composizione e organizzazione dei sistemi informatici
- Introduzione alla programmazione
- Fondamenti della programmazione in linguaggio C
- Fondamenti della programmazione in Matlab
- Argomenti avanzati di programmazione in Matlab
- Introduzione ai sistemi operativi

Cosa Imparerete:

- Gli **elementi fondamentali** ed i **principi** che regolano il funzionamento di un **sistema informatico**
- Come **sviluppare algoritmi** per risolvere problemi
- Come **codificare** tali **algoritmi** in programmi che ne permettano l'automatizzazione.
- Le basi della **programmazione**.
- L'utilizzo del linguaggio **C** e **Matlab**
- Alcune nozioni di base sui **sistemi operativi** e sulla **codifica binaria**

Bibliografia

"*Materiale su sistemi informatici e i principi di programmazione in C per il corso di Informatica B* ", Editore: Mc Graw Hill, Anno edizione: 2016, ISBN: 9781308911731 (*)

A. Campi, E. Di Nitto, D. Loiacono, A. Morzenti, B. Spoletini, "*Introduzione alla programmazione in Matlab*", seconda edizione.

Bibliografia (nota importante)

Questo testo contiene i capitoli del "*Informatica Arte e Mestiere*“ che sono rilevanti per il corso di Informatica B. Se siete in possesso di "*Informatica Arte e Mestiere*“, non dovete acquistare anche questo.

La Pagina del Corso

La pagina del corso è

http://home.deib.polimi.it/boracchi/teaching/InfoB.htm

Troverete:

- Materiale didattico usato a lezione (queste slides sono da considerare **un supporto** allo studio)
- Link ai siti delle esercitazioni e laboratori
- Temi d'esame con soluzioni
- Calendario del corso (lezioni, esercitazioni, laboratori)
- Avvisi, esiti esami/prove intermedia

N.B. Le slides caricate prima della lezione non contengono le soluzioni agli esercizi che affronteremo in aula. **Le slides vengono completate e aggiornate dopo la lezione.**

I Laboratori

- Nei laboratori vi sarà richiesto di **sviluppare autonomamente** gli elaborati.
- Tutti gli studenti nello stesso orario (un solo turno)
- Sarete assistiti da :
  - Due responsabili di laboratorio
  - Due/tre tutors
- Il Laboratorio è **molto utile** per
  - prendere familiarità con l'ambiente di sviluppo
  - consolidare la conoscenza dei linguaggi, dei metodi e degli strumenti introdotti a lezione.

I Laboratori

È possibile utilizzare il **proprio laptop**,

- Installare Code::Blocks per il C, versione con compilatore (http://www.codeblocks.org/)
- Installare Matlab (per il secondo emisemestre)
- Sul sito del laboratorio troverete tutte le istruzioni per assistervi nell'installazione

In alternativa, **vi verrà fornito un laptop** per la durata del laboratorio.

- In tal caso occorre arrivare 15 minuti prima in laboratorio per richiederlo.

Il **primo laboratorio** si terrà in quest'aula Lunedì 2 Ottobre!

L'Esame ed il Ricevimento Studenti

Modalità di Verifica

Cambia la modalità d'esame rispetto agli anni precedenti

Solo appelli regolari (niente compitini):

- **Esame scritto** su tutto il programma
- **Occorre un punteggio minimo in C ed in Matlab**
- **Non è previsto esame orale,** se non a discrezione del docente
- Il laboratorio non sarà valutato

Modalità di Verifica

Sul sito trovate **temi d'esame svolti (TDE)** degli anni precedenti

- Quest'anno la struttura del TDE cambierà leggermente
- **Durante il TDE non sarà possibile consultare libri ed appunti**
- Forniremo con il TDE dei richiami di sintassi per la programmazione in C ed in Matlab

Ricevimento Studenti

Domande e richieste di chiarimenti sono sempre ben accette.

Potete anche rivolgere domande via mail.

Però, per facilitare il lavoro di tutti, è bene evitare richieste del tipo: « è corretto così?»

Ricevimento Studenti

Domande e richieste di chiarimenti sono sempre ben accette.

Potete anche rivolgere domande via mail.

Però, per facilitare il lavoro di tutti, è bene evitare richieste del tipo: « è corretto così?»

In particolare:

- Inviate solo codici (C o Matlab) in file sorgenti
- Dite chiaramente qual è il vostro problema e perché l'esercizio non funziona
- Riportate il testo della domanda

Cos’è l’informatica?

Scienza della rappresentazione e dell’elaborazione dell’informazione.

Cos'è l'Informatica?

Scienza della rappresentazione e dell'elaborazione dell'informazione.

- ***Scienza:*** ovvero una **conoscenza sistematica** e **rigorosa** di tecniche e metodi.
- ***Informazione:*** l'oggetto dell'investigazione scientifica (informazione intesa sia come entità astratta che come tecnologie per la sua gestione)
- ***Rappresentazione:*** il modo in cui l'informazione viene strutturata e trasformata in dati fruibili da macchine
- ***Elaborazione:*** uso e trasformazione dell'informazione per un dato scopo. L'elaborazione deve poter essere eseguita da macchine che processano dati.

[da «Informatica Arte e Mestiere»]

Cos'è l'Informatica?

Scienza della rappresentazione e dell'elaborazione dell'informazione.

- ***Scienza:*** ovvero una **conoscenza sistematica** e **rigorosa** di tecniche e metodi.
- ***Informazione:*** l'oggetto dell'investigazione scientifica (informazione intesa sia come entità astratta che come tecnologie per la sua gestione)
- ***Rappresentazione:*** il modo in cui l'informazione viene strutturata e trasformata in dati fruibili da macchine
- ***Elaborazione:*** uso e trasformazione dell'informazione per un dato scopo. L'elaborazione deve poter essere eseguita da macchine che processano dati.

[da «Informatica Arte e Mestiere»]

Cos'è l'Informatica?

*Studio sistematico degli **algoritmi** che descrivono e trasformano l'informazione:*

- la loro teoria,
- analisi,
- progetto,
- efficienza,
- realizzazione,
- applicazione.

[da Association for Computing Machinery (ACM)]

Gli Algoritmi

Algoritmo

Sequenza *precisa di* ***operazioni****, definiti con* ***precisione****, che portano alla* ***realizzazione di un compito***

Le operazioni devono:

- essere **comprensibili** senza ambiguità
- essere **eseguibili** da uno strumento automatico: l'esecutore
- portare a realizzare un compito in **tempo finito** (devono contenere un numero finito di passi, ciascuno eseguibile in tempo finito)

Non è necessario un computer per fare algoritmi!

…ora vedermo un esempio di algoritmo in cui vi sarà capitato di essere esecutori

Il Linguaggio «IKEA»

- Le istruzioni IKEA sono fatte per esecutori intelligenti (noi, *ndr*) l'interpretazione dei disegni richiede diverse capacità
- Quando l'esecutore è meno intelligente occorre esprimere le istruzioni in un linguaggio più preciso

Un nostro linguaggio per gli algoritmi!

Svilupperemo i prossimi algoritmi in italiano (utilizzando un linguaggio molto essenziale).

In particolare, il linguaggio sarà caratterizzato da:

- Sequenzialità delle istruzioni
- Costrutto condizionale
- Costrutto iterative

Inoltre, potremo utilizzare “foglietti” dove scrivere (registrare) alcuni valori

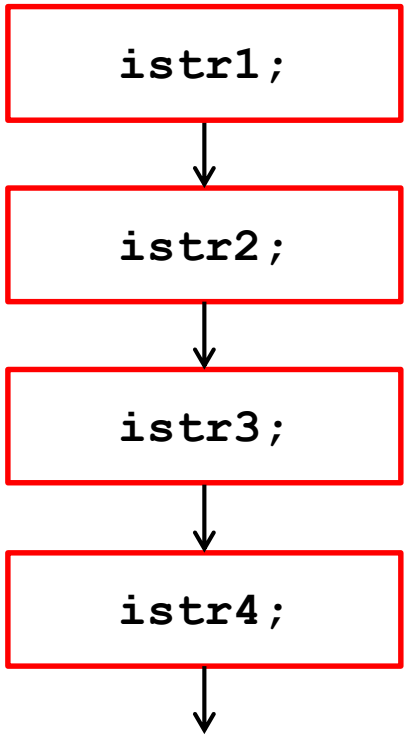
Un linguaggio più formale vi sarà presentata Venerdì ad esercitazione

La Sequenzialità

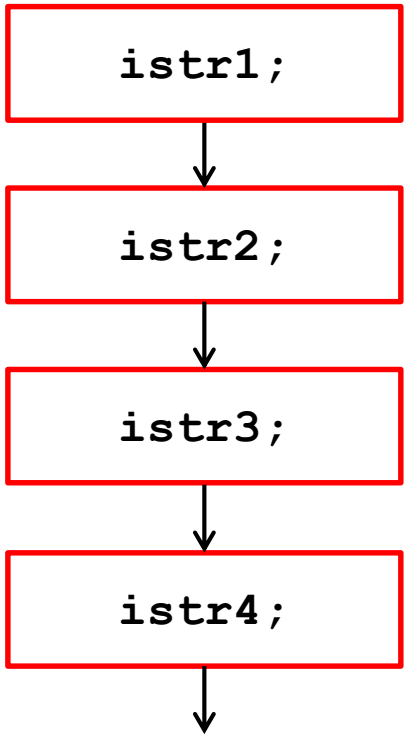
Esempio: algoritmo per andare in università

- Mi alzo quando suona la sveglia
- Mi dirigo verso la cucina
- Mangio e bevo un caffè
- Mi lavo
- Mi vesto
- Esco
- Corro

La sequenzialità



La sequenzialità




```
istr1;
istr2;
istr3;
istr4;
...
```

Le istruzioni vengono eseguite dalla prima all'ultima.

Terminata la *i-sima* istruzione, si esegue la *(i+1)-sima*

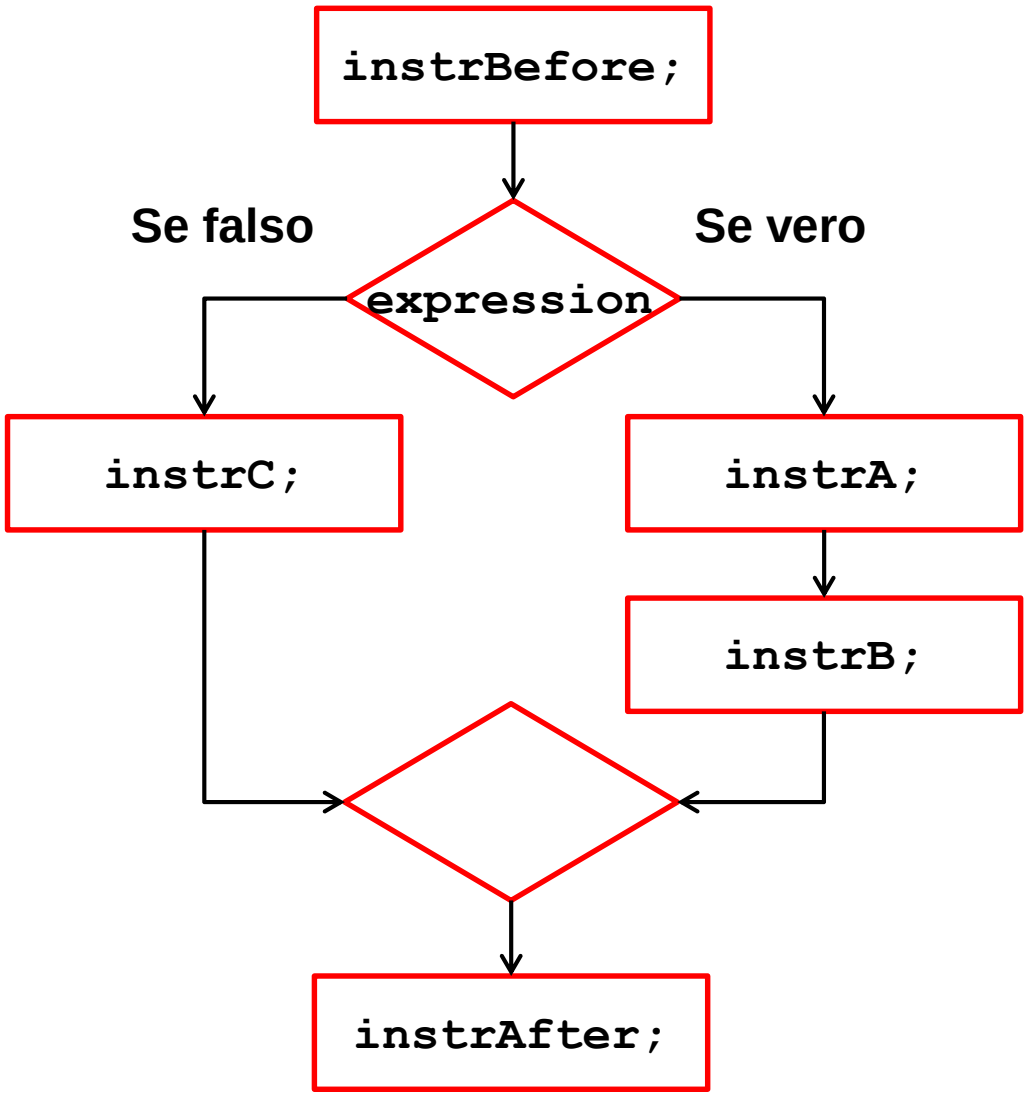
Costrutto Condizionale

Esempio: algoritmo per andare in Università

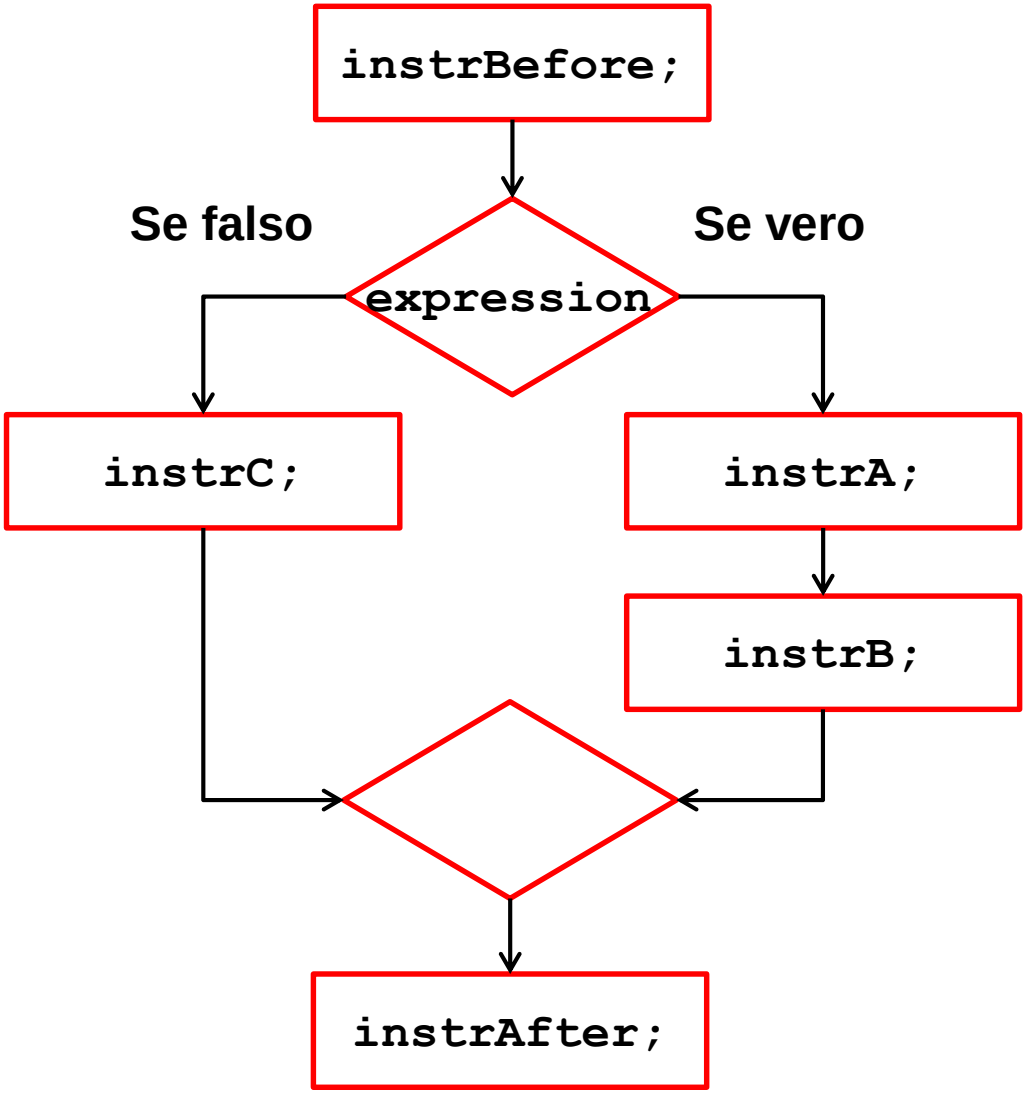
- Mi alzo quando suona la sveglia
- Mi dirigo verso la cucina
- Mangio e bevo un caffè
- Mi lavo e mi vesto
- **Se** devo mangiare fuori
  - prendo il pranzo

  Altrimenti
  - prendo uno snack per metà mattina
- Esco
- Corro

Costrutto Condizionale:



Costrutto Condizionale:

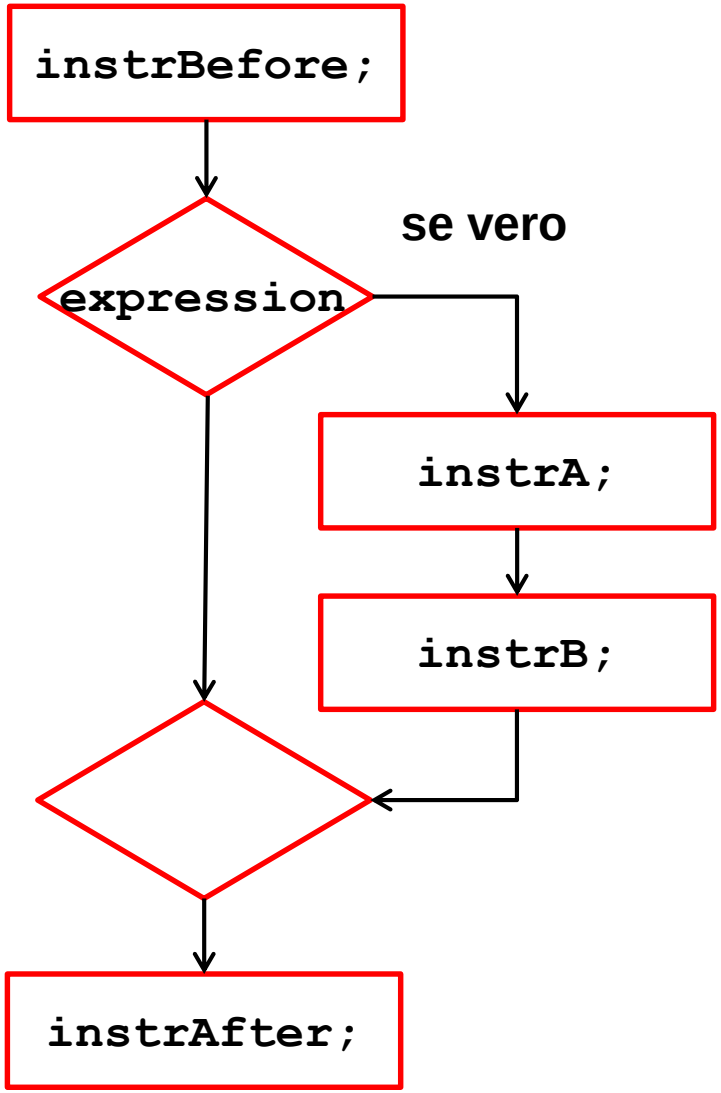


Dopo aver eseguito **`instrBefore`** si valuta **`expression`**

Se **`expression`** è vera eseguo il ramo di istruzioni contenente **`instrA;`** e **`instrB`**

Altrimenti eseguo **`instrC;`**

Costrutto Condizionale:



Non è necessario prevedere azioni specifiche nel caso in cui **`expression`** fosse falsa

Esempio: algoritmo per andare in Università

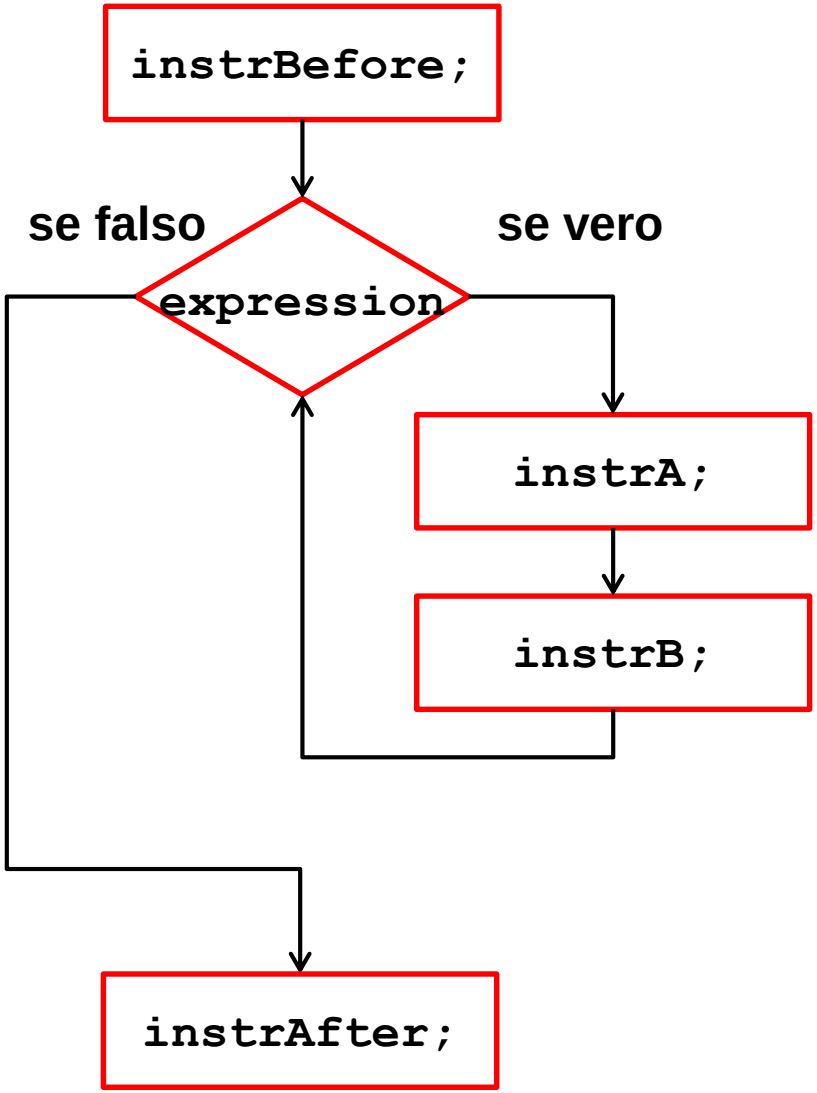
- Mi alzo quando suona la sveglia
- Mi dirigo verso la cucina
- Mangio e bevo un caffè
- Mi lavo e mi vesto
- **Se** c'è il laboratorio di informatica
  - Prendo il laptop
- Esco
- Corro

Costrutto Iterativo

Esempio: algoritmo per andare in Università

- Mi alzo quando suona la sveglia
- Mi dirigo verso la cucina
- Mangio e bevo un caffè
- Mi lavo e mi vesto
- **Se** c'è il laboratorio di informatica
  - Prendo il laptop
- Vado in stazione
- **Ripeti**: "attendi un treno"
  - Se ferma a Bovisa, sali sul treno
- Arrivi a lezione, wow

Costrutto Iterativo



Se **`expression`** è vera eseguo il ramo di istruzioni contenente **`instrA;`** e **`instrB;`** (corpo del ciclo)

Al termine valuta nuovamente **`expression`**.

Se **`expression`** è falsa, prosegui oltre, altrimenti esegui le istruzioni nel corpo del ciclo

Esempio: algoritmo per andare in Università

- Mi alzo quando suona la sveglia
- Mi dirigo verso la cucina
- Mangio e bevo un caffè
- Mi lavo e mi vesto
- **Se** c'è il laboratorio di informatica
  - Prendo il laptop
- Vado in stazione
- **Ripeti**: "guarda il treno in arrivo: non ferma a Bovisa?"
  - se vero, attendi il possimo treno
- Sali sul treno
- Arrivi a lezione, wow

Esempi di Algoritmi

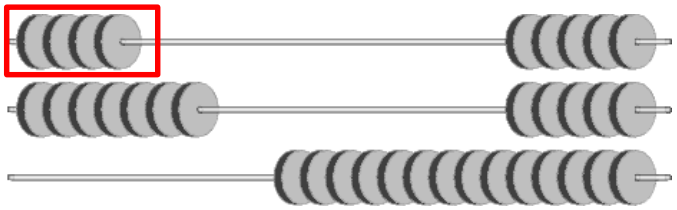
Il Pallottoliere

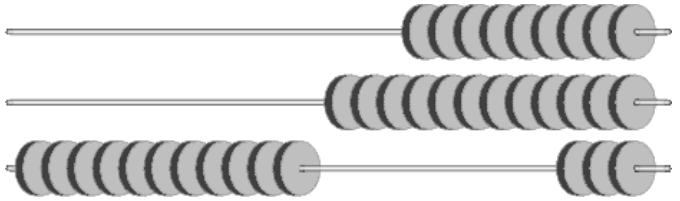
Supponiamo di avere un bambino che sa contare, ma non sa fare operazioni aritmetiche e non sa scrivere. Scriviamo le istruzioni per utilizzare il pallottoliere

Algoritmo per sommare col pallottoliere

Per semplificarci la vita suppoiamo che :

- Primo addendo è riportato sulla prima riga a sinistra
- Secondo addendo è sulla seconda riga a sinistra
- Risultato deve apparier nella terza riga
- Operiamo con numeri «piccoli», non c'è bisogno del riporto

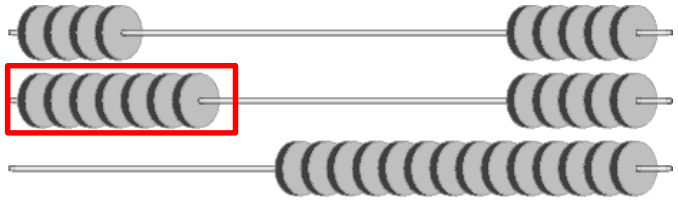


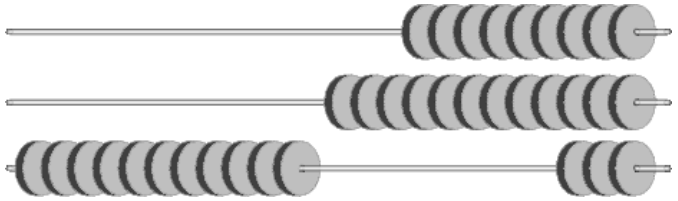


Algoritmo per sommare col pallottoliere

Per semplificarci la vita suppoiamo che :

- Primo addendo è riportato sulla prima riga a sinistra
- Secondo addendo è sulla seconda riga a sinistra
- Risultato deve apparier nella terza riga
- Operiamo con numeri «piccoli», non c'è bisogno del riporto

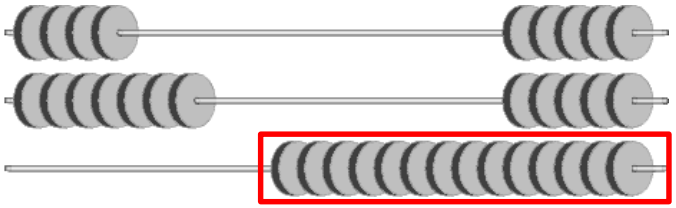


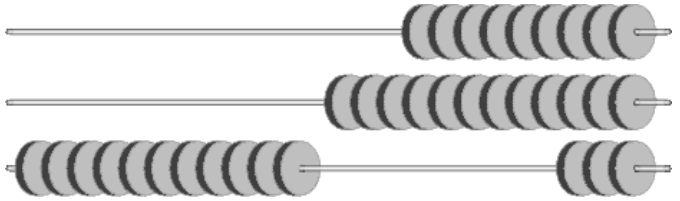


Algoritmo per sommare col pallottoliere

Per semplificarci la vita suppoiamo che :

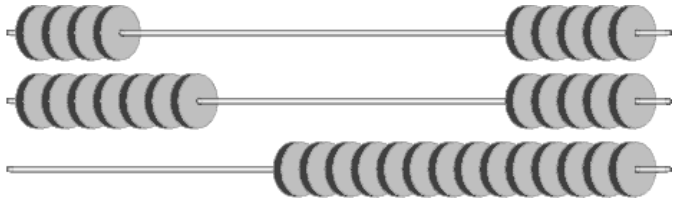
- Primo addendo è riportato sulla prima riga a sinistra
- Secondo addendo è sulla seconda riga a sinistra
- Risultato deve apparier nella terza riga
- Operiamo con numeri «piccoli», non c'è bisogno del riporto

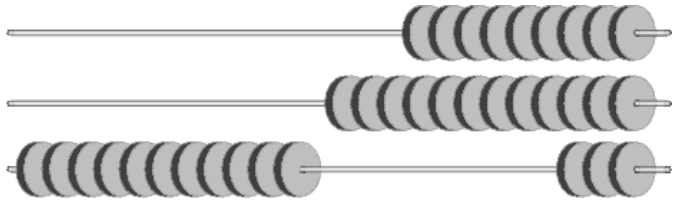




Algoritmo del pallottoliere

1. Sposta una pallina da sinistra a destra nella prima riga, al contempo da destra a sinistra nella terza riga
2. **Ripeti** operazione precedente **fino a** svuotare parte sinistra prima riga
3. Sposta pallina da sinistra a destra nella seconda riga, al contempo da destra a sinistra nella terza.
4. **Ripeti** operazione precedente **fino a** svuotare parte sinistra seconda riga
5. Conta quante palline trovi sulla terza riga.





Proprietà fondamentali degli algoritmi

Correttezza:

- l'algoritmo risolve il compito senza errori o difetti.

Efficienza:

- l'algoritmo usa risorse in modo minimale (o almeno ragionevole)

Altri Esempi:

- Algoritmo per invertire il contenuto di due bicchieri, supponendo di avere un terzo bicchiere.
- Algoritmo per trovare il prodotto più buono nella corsia del supermercato, passando per la corsia una sola volta.
- Algoritmo per controllare che la maggioranza degli studenti abbia capito gli esercizi sopra.
- Algoritmo per ricercare i libri in Biblioteca

Altri Esempi:

- Algoritmo per invertire il contenuto di due bicchieri, supponendo di avere un terzo bicchiere.

- Algoritmo per trovare il prodotto più buono nella corsia del supermercato, passando per la corsia una sola volta.

- Algoritmo per controllare che la maggioranza degli studenti abbia capito gli esercizi sopra.

- Algoritmo per ricercare i libri in Biblioteca

Esempio: invertire il contenuto di A e B

1. Prendi un terzo bicchiere C
2. Rovescia il contenuto del bicchiere A nel bicchiere C
3. Rovescia il contenuto di B in A
4. Rovescia il contenuto di C in B

Esempio: invertire il contenuto di A e B

1. Prendi un terzo bicchiere C
2. Rovescia il contenuto del bicchiere A nel bicchiere C
3. Rovescia il contenuto di B in A
4. Rovescia il contenuto di C in B

Algoritmo per scambiare i valori di due variabili A e B (con le variabili a volte non occorre C)

Altri Esempi:

- Algoritmo per invertire il contenuto di due bicchieri, supponendo di avere un terzo bicchiere.
- Algoritmo per trovare il prodotto più buono nella corsia del supermercato, passando per la corsia una sola volta.
- Algoritmo per controllare che la maggioranza degli studenti abbia capito gli esercizi sopra.
- Algoritmo per ricercare i libri in Biblioteca

Esempio: ricerca del prodotto migliore

1. Prendi in mano il primo prodotto: assumi che sia il migliore
2. Procedi fino al prossimo prodotto
3. Confrontalo con quello che hai in mano
4. **Se** il prodotto davanti a te è migliore: abbandona il prodotto che hai in mano e prendi quello sullo scaffale
5. **Ripeti** i passi **2 - 4** **fino a** raggiungere la fine della corsia
6. Hai in mano il prodotto migliore.

Esempio: ricerca del prodotto migliore

1. Prendi in mano il primo prodotto: assumi che sia il migliore
2. Procedi fino al prossimo prodotto
3. Confrontalo con quello che hai in mano
4. **Se** il prodotto davanti a te è migliore: abbandona il prodotto che hai in mano e prendi quello sullo scaffale
5. **Ripeti** i passi **2 - 4** **fino a** raggiungere la fine della corsia
6. Hai in mano il prodotto migliore.

Algoritmo per trovare il massimo di una sequenza numerica

Altri Esempi:

- Algoritmo per invertire il contenuto di due bicchieri, supponendo di avere un terzo bicchiere.
- Algoritmo per trovare il prodotto più buono nella corsia del supermercato, passando per la corsia una sola volta.
- Algoritmo per controllare che la maggioranza degli studenti abbia capito gli esercizi sopra.
- Algoritmo per ricercare i libri in Biblioteca

Esempio: assicurarsi che il 50% abbia capito

1. Prendi due **fogli**, uno per contare chi non ha capito (N) ed uno per contare tutti gli studenti (T)
2. Avvicinati al primo studente
3. Chiedi se ha capito
4. **Se** risponde no, metti un segno su N
5. Metti un segno su T
6. Passa al prossimo studente
7. **Ripeti** i passi **3 – 6 fino** all'ultimo studente
8. Conta i segni su N e su T
9. **Se** il numero di N è maggiore della metà di T, la condizione è non verificata

Esempio: assicurarsi che tutti abbiano capito

1. Prendi un solo **foglio**
2. Avvicinati al primo studente
3. Chiedi se ha capito
4. **Se** risponde no, metti un segno sul foglio
5. Passa al prossimo studente
6. **Ripeti** i passi **3 – 5 fino** all'ultimo studente
7. Se il foglio non ha segni, tutti hanno capito.

Esempio: assicurarsi che tutti abbiano capito

1. Prendi un solo **foglio**
2. Avvicinati al primo studente
3. Chiedi se ha capito
4. **Se** risponde no, metti un segno sul foglio
5. Passa al prossimo studente
6. **Ripeti** i passi **3 – 5 fino** all'ultimo studente **o fino** a quando uno studente dice di non aver capito
7. Se il foglio non ha segni, tutti hanno capito.

Variante più efficiente

Esempio: assicurarsi che tutti abbiano capito

1. Prendi un solo **foglio**
2. Avvicinati al primo studente
3. Chiedi se ha capito
4. **Se** risponde no, metti un segno sul foglio
5. Passa al prossimo studente
6. **Ripeti** i passi **3 – 5 fino** all'ultimo studente **o fino** a quando uno studente dice di non aver capito
7. Se il foglio non ha segni, tutti hanno capito.

Algoritmo per verificare che una condizione sia soddisfatta da tutti gli elementi di un array

Altri Esempi:

- Algoritmo per invertire il contenuto di due bicchieri, supponendo di avere un terzo bicchiere.
- Algoritmo per trovare il prodotto più buono nella corsia del supermercato, passando per la corsia una sola volta.
- Algoritmo per controllare che la maggioranza degli studenti abbia capito gli esercizi sopra.
- Algoritmo per ricercare i libri in Biblioteca

Gestione di una biblioteca

- Si supponga di avere una biblioteca gestita mediante archivio cartaceo.
- Ogni libro ha una data posizione identificata da SCAFFALE e POSIZIONE
- Esiste un archivio ordinato che contiene, per ogni libro un foglio del tipo:

AUTORE / I:
GHEZZI CARLO,
JAZAYERI MEHDI.

TITOLO:
PROGRAMMING LANGUAGE CONCEPTS.
1981.

SCAFFALE 35
POSIZIONE 21

Gestione di una biblioteca

Algoritmo per trovare un libro

1. **ricerca** la scheda del libro nello schedario
2. trovata la scheda, **segna** su un **foglio** numero scaffale e posizione del libro
3. raggiungi lo scaffale indicato
4. individuato lo scaffale, **ricerca** la posizione del libro
5. Prendi il libro

AUTORE / I:
GHEZZI CARLO,
JAZAYERI MEHDI.

TITOLO:
PROGRAMMING LANGUAGE CONCEPTS.
1981.

SCAFFALE 35
POSIZIONE 21

Algoritmo: Ricerca

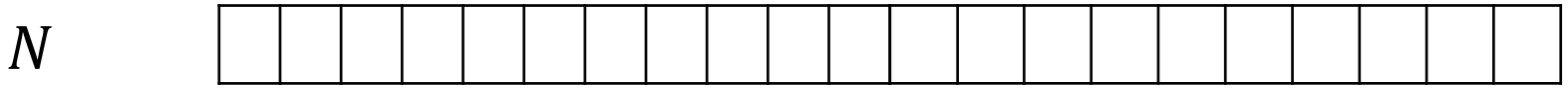
Non tutti gli esecutori sono in grado di «cercare», e comunque la ricerca può essere fatta in diversi modi

1. esamina la **prima** scheda dello schedario
2. **se** autore e titolo coincidono con quelli cercati
   - ricerca **conclusa** con successo

   altrimenti passa a scheda successiva
3. **Ripeti** istruzione **2**, **fino a conclusione o fino a raggiungere l'ultima scheda**
4. **se** trovata $\Rightarrow$ ricerca conclusa con successo
   - **altrimenti** $\Rightarrow$ ricerca conclusa con insuccesso

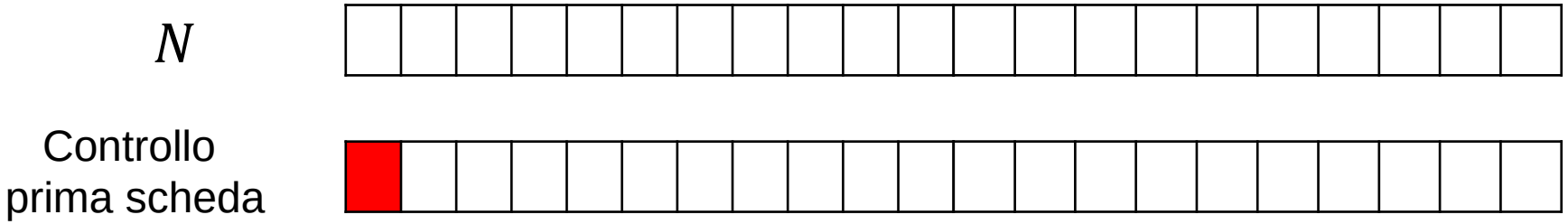
Algoritmo: Ricerca

Assumendo che l'archivio abbia N schede:



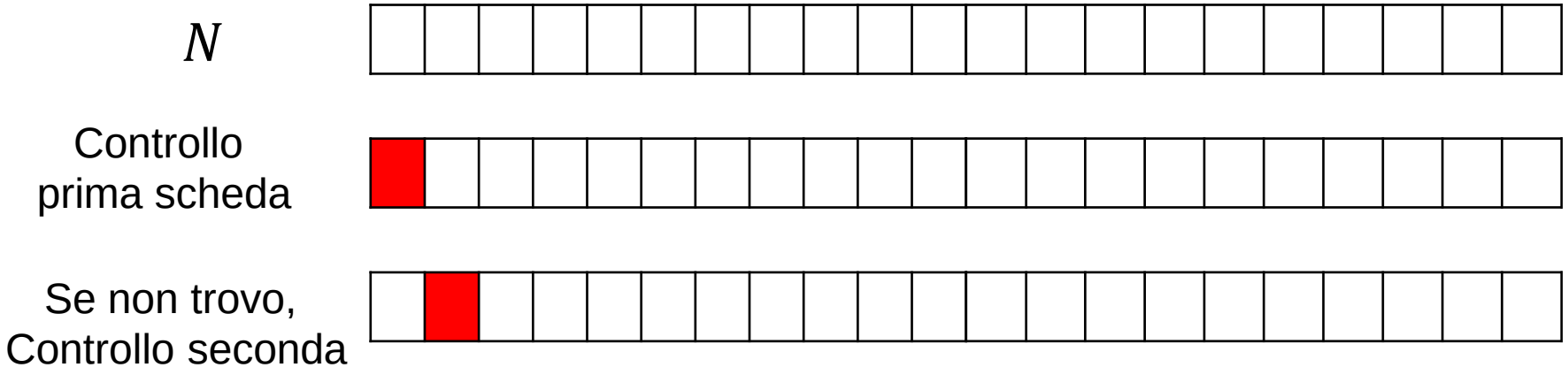
Algoritmo: Ricerca

Assumendo che l'archivio abbia N schede:



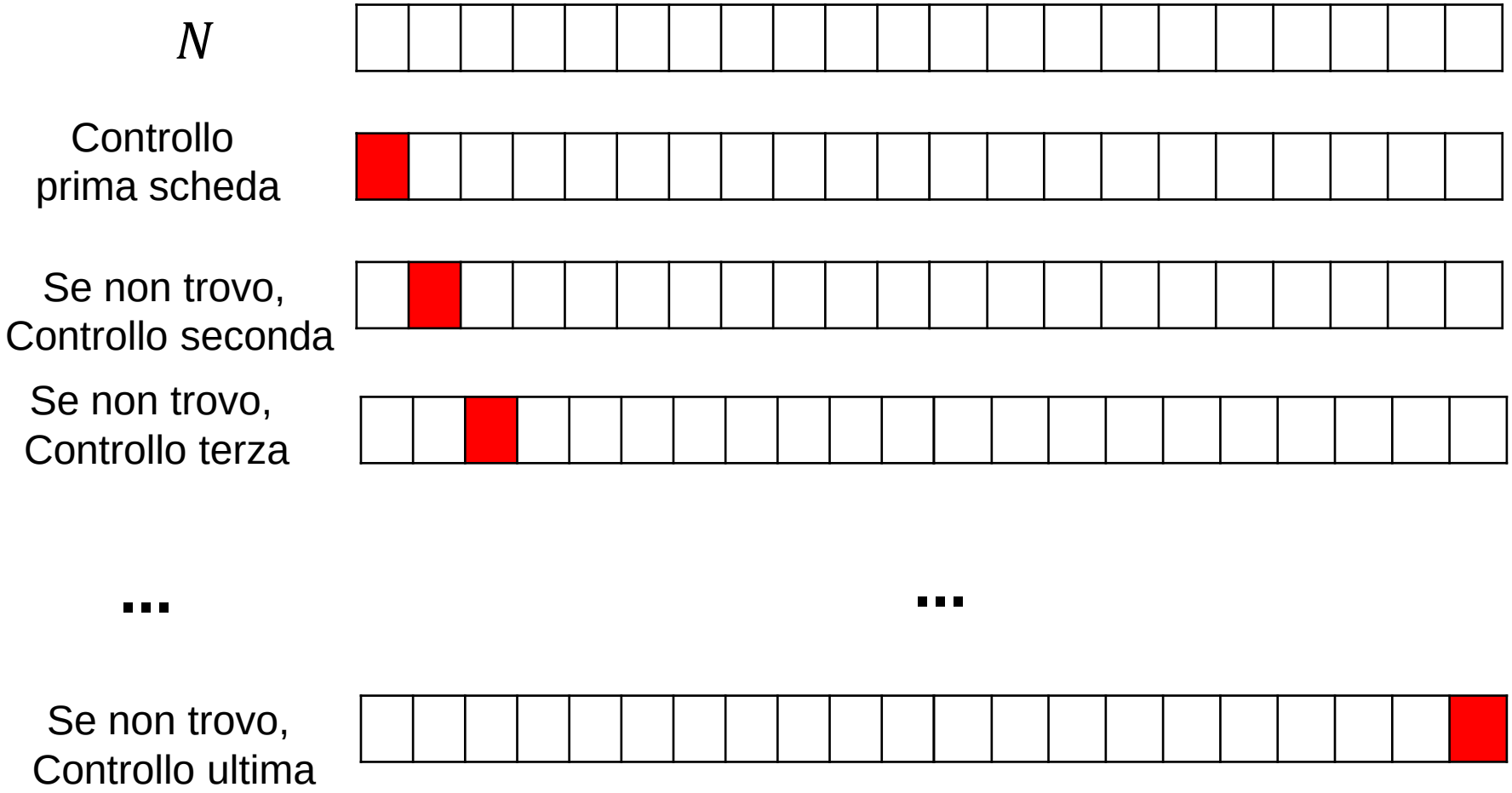
Algoritmo: Ricerca

Assumendo che l'archivio abbia N schede:



Algoritmo: Ricerca

Assumendo che l'archivio abbia N schede:



Algoritmo: Ricerca

Non tutti gli esecutori sono in grado di «cercare», e comunque la ricerca può essere fatta in diversi modi

1. esamina la **prima** scheda dello schedario
2. **se** autore e titolo coincidono con quelli cercati
   - ricerca **conclusa** con successo

   altrimenti passa a scheda successiva
3. **Ripeti** istruzione **2**, **fino a conclusione o fino a raggiungere l'ultima scheda**
4. **se** trovata $\Rightarrow$ ricerca conclusa con successo
   - **altrimenti** $\Rightarrow$ ricerca conclusa con insuccesso

Ricerca semplice ma **inefficiente**: non sfrutta l'ordinamento delle schede nell'archivio

Algoritmo: Ricerca tra elementi ordinati

Lo schedario contiene N schede ordinate in cui cercare

1. prendi la scheda centrale, i.e. alla posizione $N/2$.
2. **se** è la scheda cercata, termina con successo.
   altrimenti,
3. **se** la scheda cercata segue alfabeticamente quella esaminata,
   - continua la **ricerca** nella seconda metà dello schedario
   - **altrimenti** continua la **ricerca** nella prima metà

Algoritmo: Ricerca tra elementi ordinati

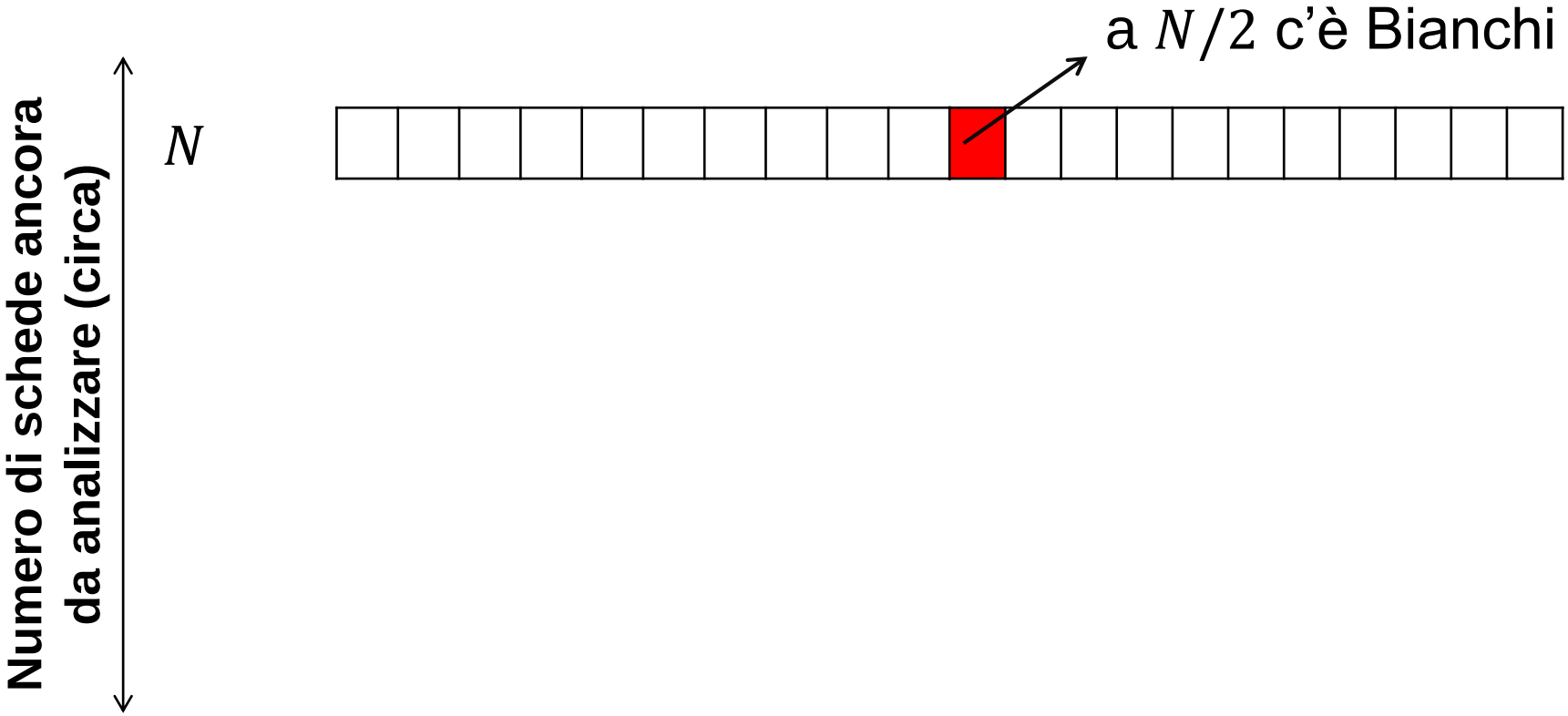
Lo schedario contiene N schede ordinate in cui cercare

1. prendi la scheda centrale, i.e. alla posizione $N/2$.
2. **se** è la scheda cercata, termina con successo.
   altrimenti,
3. **se** la scheda cercata segue alfabeticamente quella esaminata,
   - continua la **ricerca** nella seconda metà dello schedario
   - **altrimenti** continua la **ricerca** nella prima metà

Questo algoritmo ricerca **è ricorsivo, perché** richiama se stesso (riduce però il numero di schede da analizzare ed esiste una condizione per cui non chiama se stesso).

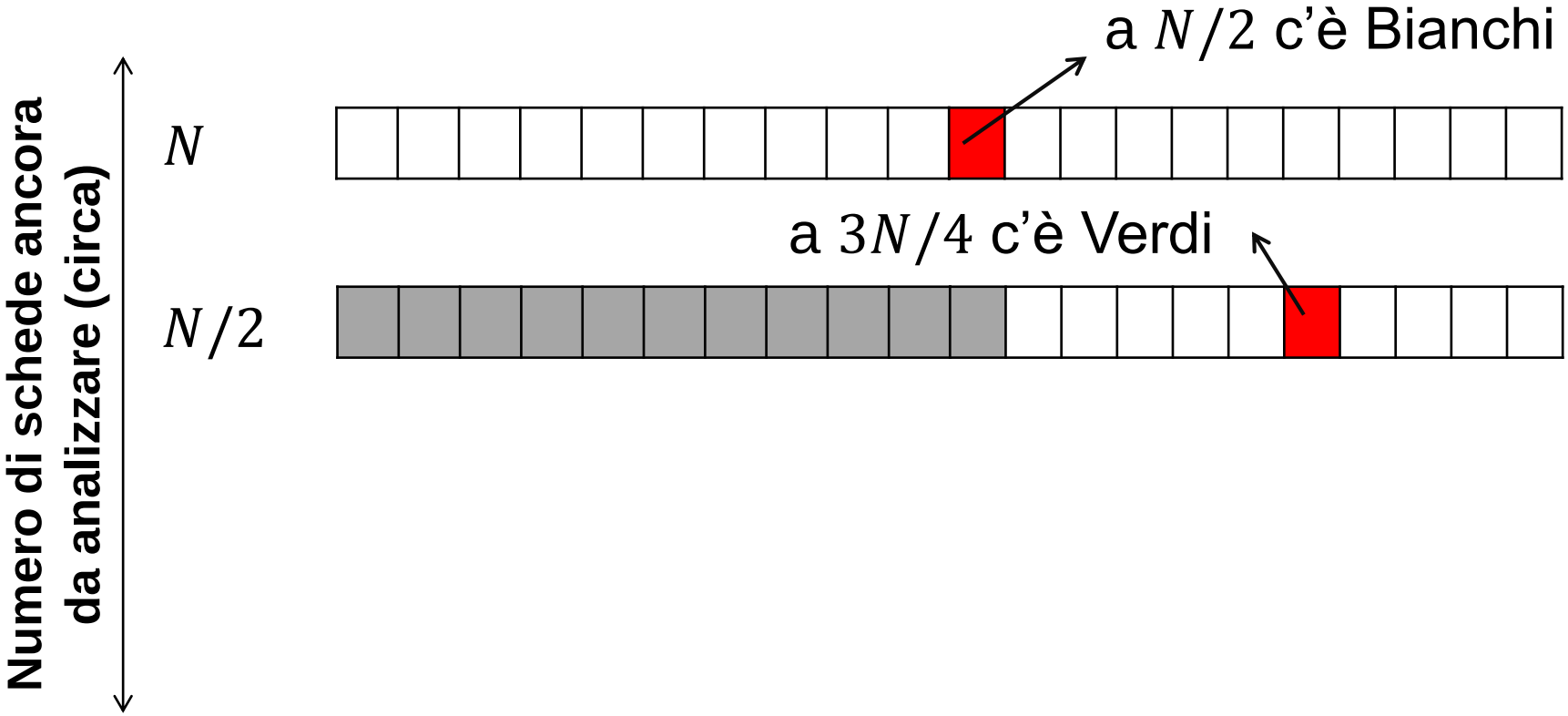
Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



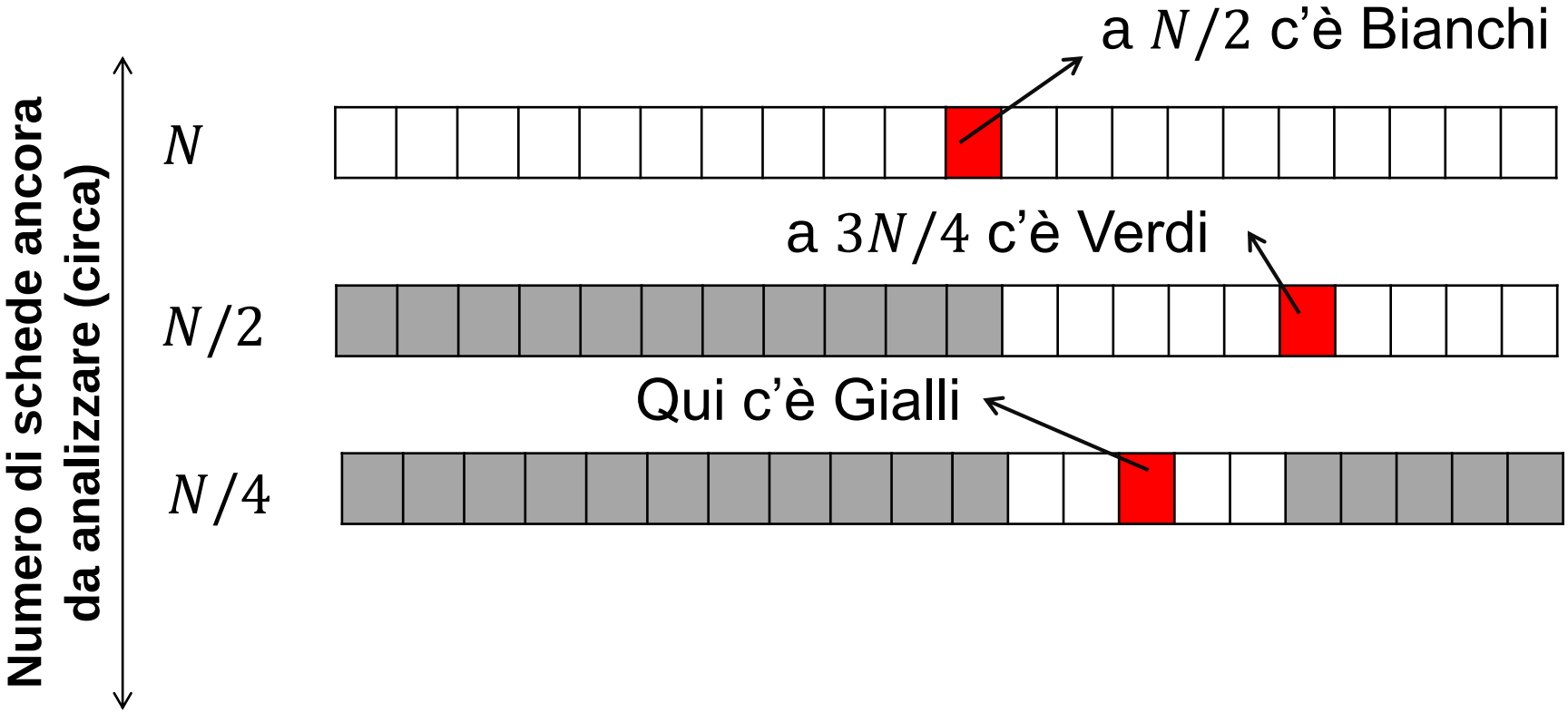
Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



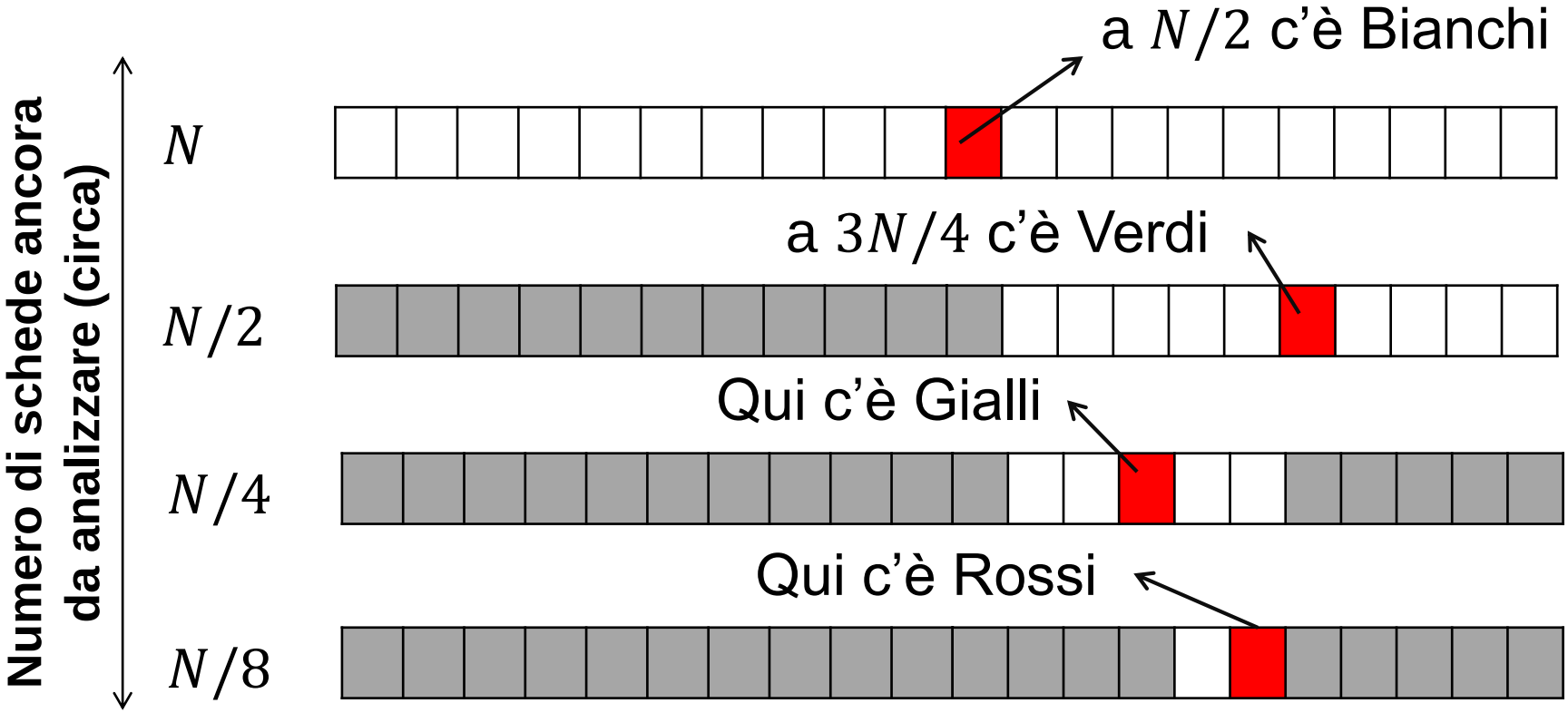
Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



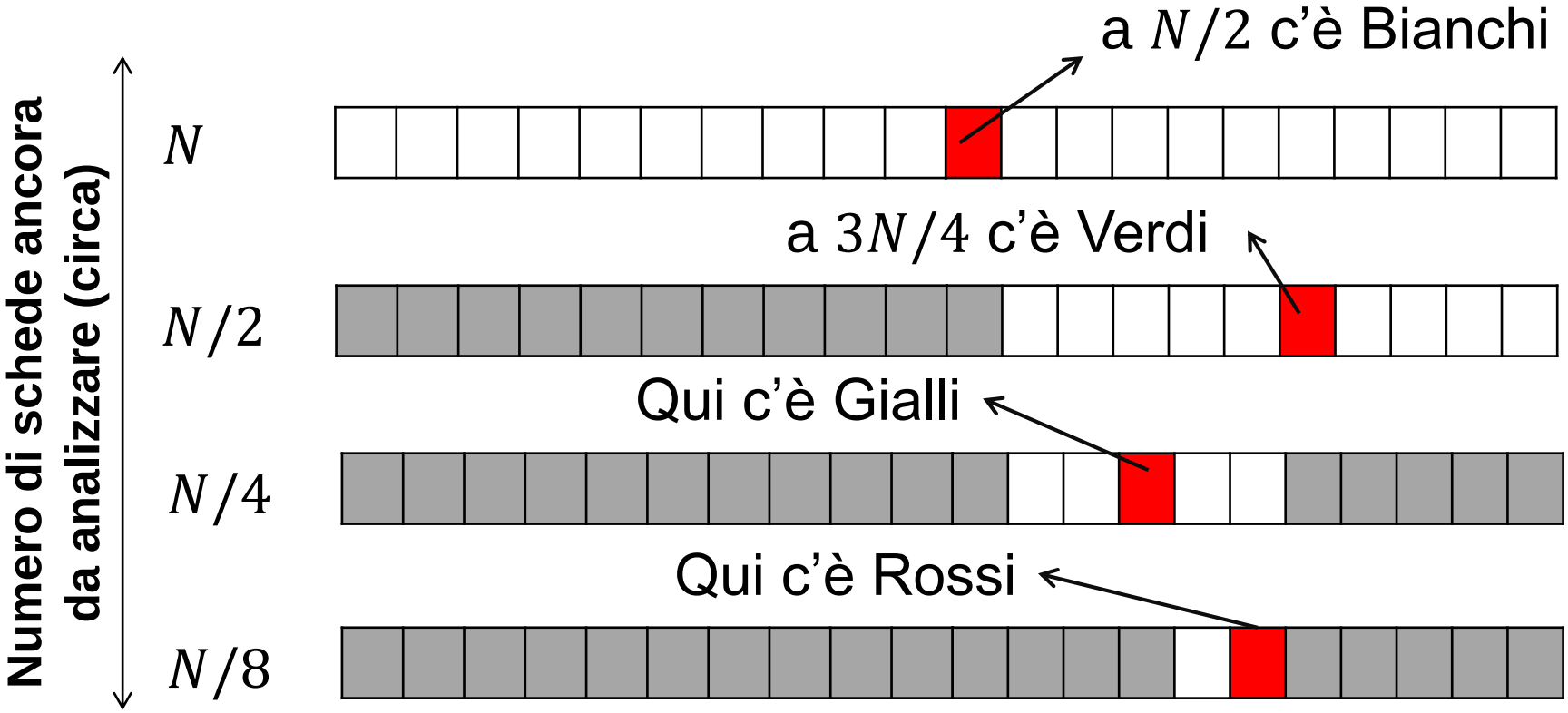
Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



N.B: cosa succede se non esiste il nome cercato nell'archivio? L'algoritmo deve terminare anche in questo caso!

Algoritmo: Ricerca tra elementi ordinati

Lo schedario contiene N schede ordinate in cui cercare

1. prendi la scheda centrale, i.e. alla posizione $N/2$.
2. **se** è la scheda cercata, termina con successo.
   altrimenti,
3. **se** la scheda cercata segue alfabeticamente quella esaminata,
   - continua la **ricerca** nella seconda metà dello schedario
   - **altrimenti** continua la **ricerca** nella prima metà

Cosa devo modificare per far terminare l'algoritmo quando la zona di ricerca è vuota?

Algoritmo: Ricerca tra elementi ordinati

Lo schedario contiene N schede ordinate in cui cercare

1. **se** N è zero (porzione di schedario vuota) allora termina la ricerca, **altrimenti**, prendi la scheda alla posizione $N/2$.
2. **se** è la scheda cercata termina con successo
   altrimenti,
3. **se** la scheda cercata segue alfabeticamente quella esaminata,
   - continua la **ricerca** nella seconda metà dello schedario
   - **altrimenti** continua la **ricerca** nella prima metà

I Programmi

Dall'algoritmo al codice

Il Calcolatore

Il **calcolatore** è un potente **esecutore di algoritmi**

- ↑ È rapido: permette di gestire quantità di informazioni altrimenti intrattabili
- ↑ È preciso: non commette mai errori
- ↓ Non ha spirito critico

I **programmi** sono **algoritmi codificati** in **linguaggi** comprensibili dal calcolatore.

Il Programmatore (voi)

Compito dell'informatico (e di chiunque programmi) è:

1. **Ideare l'algoritmo**: conoscere la soluzione del problema e esprimerla in rigorosi passi
2. **Codificare l'algoritmo in un programma**: conoscere il linguaggio dell'esecutore (linguaggio di programmazione)

La parte più difficile è spesso la prima.

- Prima dobbiamo aver chiaro «cosa far fare alla macchina», poi traduciamolo correttamente.

Proprietà essenziali dei programmi (algoritmi)

Correttezza: l'algoritmo risolve il compito senza errori o difetti.

Efficienza: l'algoritmo usa risorse in modo minimale/ragionevole

- **Diversi criteri** quindi per definire l'efficienza a seconda delle risorse
  - tempo,
  - memoria,
  - numero di letture/scritture da disco

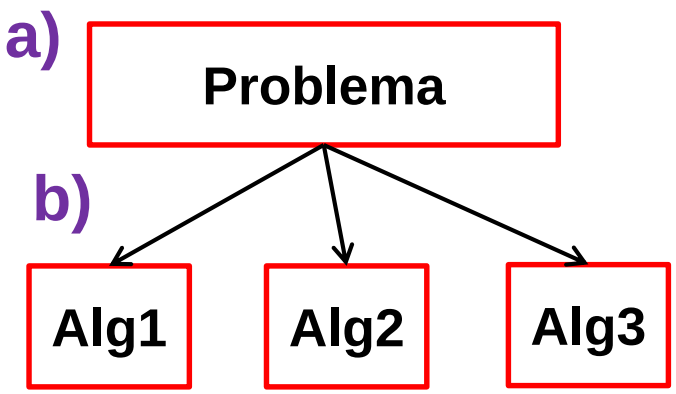
Riepilogando: Come Procedere

a) Partirete dall'analisi di un problema

Riepilogando: Come Procedere

a) Partirete dall'analisi di un problema

b) Individuerete uno o più algoritmi che risolvono il problema in modo più o meno efficiente

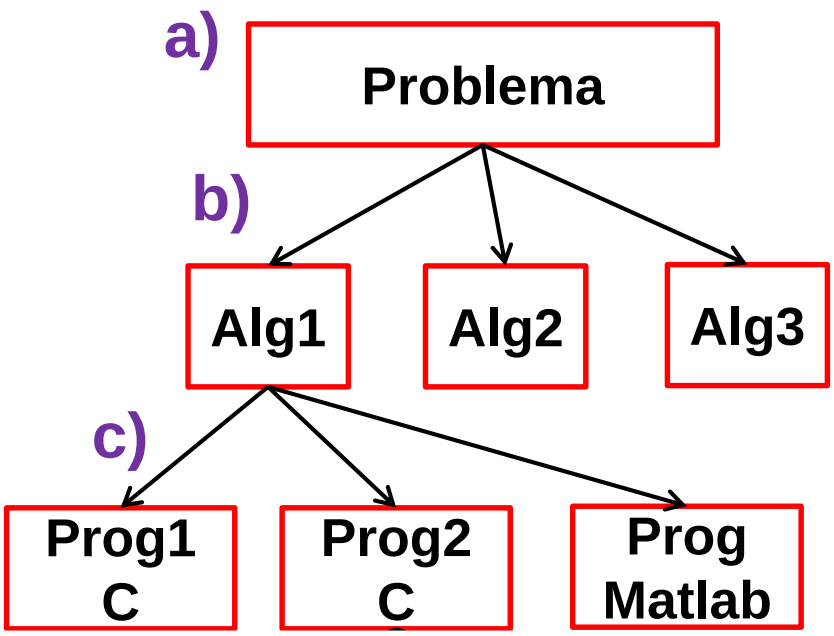


Riepilogando: Come Procedere

a) Partirete dall'analisi di un problema

b) Individuerete uno o più algoritmi che risolvono il problema in modo più o meno efficiente

c) Codificherete un algoritmo in un linguaggio di programmazione,



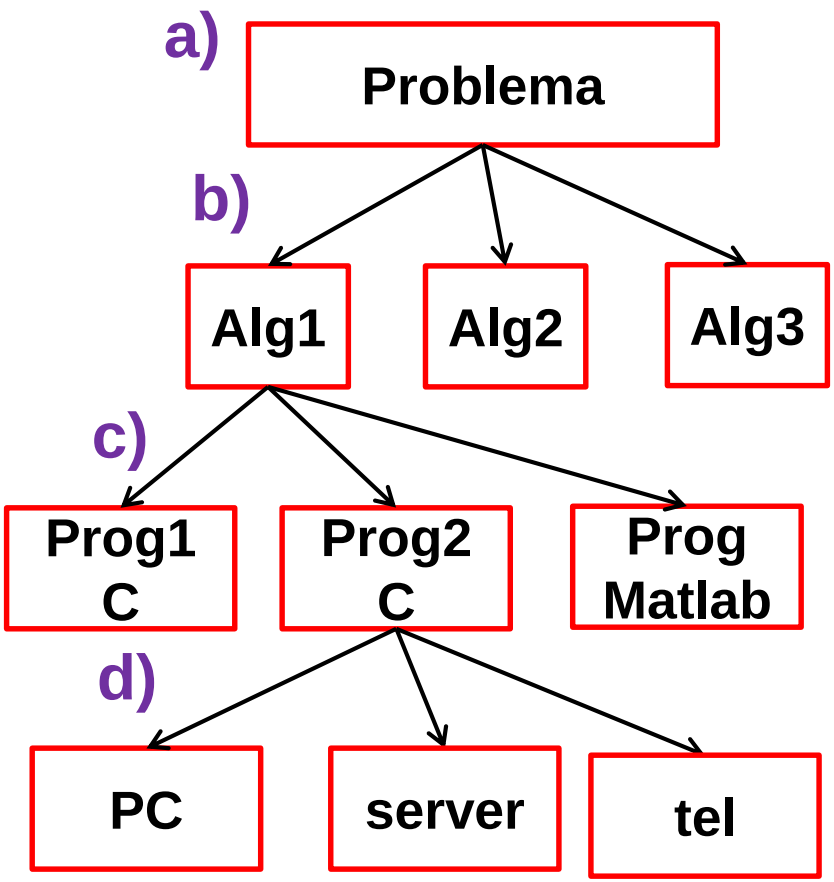
Riepilogando: Come Procedere

a) Partirete dall'analisi di un problema

b) Individuerete uno o più algoritmi che risolvono il problema in modo più o meno efficiente

c) Codificherete un algoritmo in un linguaggio di programmazione,

d) Il programma potrà essere «utilizzato» su diverse macchine



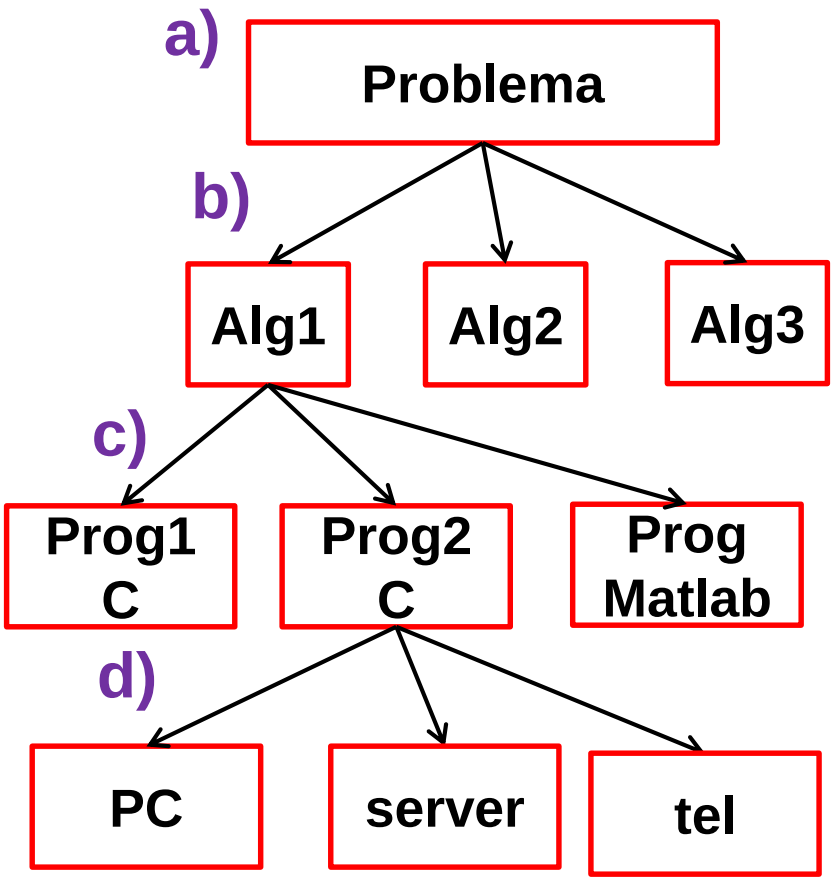
Riepilogando: Cosa Fare

a) Dovrete (capire e) definire correttamente il problema

b) **Ricavare l'algoritmo** è la parte più **difficile**. Richiede, oltre a esperienza, competenze specifiche, creatività e anche rigore perché occorre specificarlo in modo formale

c) La codifica è più semplice ma il programma deve essere efficiente e corretto

d) All'ultimo step pensa la macchina… vedremo poi la compilazione



La Rappresentazione dell'Informazione

Cos'è l'Informatica?

Scienza della rappresentazione e dell'elaborazione dell'informazione.

- ***Scienza:*** ovvero una **conoscenza sistematica** e **rigorosa** di tecniche e metodi.
- ***Informazione:*** l'oggetto dell'investigazione scientifica (informazione intesa come entità astratta e come tecnologie per la sua gestione)
- ***Rappresentazione:*** il modo in cui l'informazione viene strutturata e trasformata in dati fruibili da macchine
- ***Elaborazione:*** uso e trasformazione dell'informazione per un dato scopo. L'elaborazione deve poter essere eseguita da macchine che processano dati.

[da «Informatica Arte e Mestiere»]

Rappresentazione dell'informazione

I calcolatori sono in grado di operare con informazioni **binarie**. Gli unici simboli che possiamo utilizzare sono {0,1}

Rappresentazione dell'informazione

I calcolatori sono in grado di operare con informazioni **binarie**. Gli unici simboli che possiamo utilizzare sono $\{0,1\}$

Un **bit** (*binary digit*) assume valore 0/1 corrispondente ad un determinato *stato fisico* (alta o bassa tensione nella cella di memoria)

Il **Byte** è una sequenza di 8 bit ed esprime $2^8 = 256$ numeri diversi (ad esempio gli interi in $[0,255]$)

00000000, 00000001, 00000010, …, 11111111

Rappresentazione dell'informazione

I calcolatori sono in grado di operare con informazioni **binarie**. Gli unici simboli che possiamo utilizzare sono $\{0,1\}$

Un **bit** (*binary digit*) assume valore 0/1 corrispondente ad un determinato *stato fisico* (alta o bassa tensione nella cella di memoria)

Il **Byte** è una sequenza di 8 bit ed esprime $2^8 = 256$ numeri diversi (ad esempio gli interi in $[0,255]$)

00000000, 00000001, 00000010, …, 11111111

Codifica binaria di un numero: la sua **rappresentazione** come una **sequenza di 0 ed 1.**

In questo corso vedremo la **codifica binaria di numeri interi** e **razionali** e semplici **operazioni aritmetiche e logiche**

Aritmetica Binaria

Un po' di operazioni in base 2….

Codifica dei Numeri in Base 10

- Utilizziamo una **notazione posizionale**:
  - Le cifre all'interno di un numero hanno un significato differente a seconda della posizione

- Le **cifre** che abbiamo a disposizione sono 10

$$\{0, 1, \ldots, 9\}$$

Codifica dei Numeri in Base 10

- Utilizziamo una **notazione posizionale**:
  - Le cifre all'interno di un numero hanno un significato differente a seconda della posizione
- Le **cifre** che abbiamo a disposizione sono 10

$$\{0, 1, \ldots, 9\}$$

- Es numero di 4 cifre
  - $\mathbf{3401} = \mathbf{3} \times 10^3 + \mathbf{4} \times 10^2 + \mathbf{0} \times 10^1 + \mathbf{1} \times 10^0$

  È diverso da altre combinazioni delle stesse cifre quali:
  - $\mathbf{1403} = \mathbf{1} \times 10^3 + \mathbf{4} \times 10^2 + \mathbf{0} \times 10^1 + \mathbf{3} \times 10^0$
  - $\mathbf{0314} = \mathbf{0} \times 10^3 + \mathbf{3} \times 10^2 + \mathbf{1} \times 10^1 + \mathbf{4} \times 10^0$

Codifica dei Numeri in Base 10

- Le **cifre** sono **gli elementi** dell'alfabeto che abbiamo a disposizione per scrivere i numeri
- In base 10 abbiamo questo alfabeto:

$$A_{10} = \{0,1,\dots,9\}$$

- I **numeri** sono sequenze ordinate di cifre (es 3401)

Codifica dei Numeri in Base 10

- Le **cifre** sono **gli elementi** dell'alfabeto che abbiamo a disposizione per scrivere i numeri
- In base 10 abbiamo questo alfabeto:

$$A_{10} = \{0,1,\dots,9\}$$

- I **numeri** sono sequenze ordinate di cifre (es 3401)
- Un numero di m cifre $(3401, m = 4$, e $332, m = 3)$ si può scrivere come:

$$(N)_{10} = a_{m-1}a_{m-2}\dots a_1 a_0 = \sum_{i=0}^{m-1} a_i \times 10^i \, , a_i \in A_{10}$$

- Con m cifre posso rappresentare 10^m numeri distinti:

$$0, \dots, 10^m - 1$$

Codifica dei numeri in base 2: Esempi

Qual è il numero massimo che posso scrivere con m bit?

Codifica dei numeri in base 2: Esempi

Qual è il numero massimo che posso scrivere con m bit?

Quante combinazioni di m elementi posso realizzare se ogni elemento lo scelgo tra 2 diversi? -> 2^m

Codifica dei numeri in base 2: Esempi

Qual è il numero massimo che posso scrivere con m bit?

Quante combinazioni di m elementi posso realizzare se ogni elemento lo scelgo tra 2 diversi? -> 2^m

Ad esempio:

Con 1 cifra in base 2, copro $[0, 2^1 - 1]$ (cioè $[0, 1]$)

Con 4 cifre in base 2, copro $[0, 2^4 - 1]$ (cioè $[0, 15]$)

Con 8 cifre in base 2, copro $[0, 2^8 - 1]$ (cioè $[0, 255]$)

Con 16 cifre in base 2, copro $[0, 2^{16} - 1]$ (cioè $[0, 65535]$)

In binario ho bisogno di molte più cifre rispetto al decimale per esprimere gli stessi numeri

Conversione Binario-Decimale

Utilizziamo la definizione di numero in notazione posizionale

$$(N)_2 = a_{m-1} \times 2^{m-1} + a_{m-2} \times 2^{m-2} + .. + a_0 \times 2^0$$

Es.

$$(101)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = (5)_{10}$$

$$(1100010)_2 = 1 \times 2^6 + 1 \times 2^5 + 1 \times 2 =$$

$$= 64 + 32 + 2 = (98)_{10}$$

Nel corso vedremo diverse codifiche per

- I numeri interi (positivi e negativi)
- I numeri razionali

ed inoltre vedremo come eseguire la somma di interi

… e tutte le altre potenze di 2?

È necessario imparare le potenze di 2!

2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
1	2	4	8	16	32	64	128	**256**	512	1024

E i loro legami con l'informatica:

- Byte = 8 bit
- KiloByte (kB) = 10^3 Byte
- MegaByte (MB) = 10^6 Byte
- GigaByte (GB) = 10^9 Byte
- TheraByte (TB) = 10^{12} Byte

Rappresentazione dei Caratteri

Ogni carattere viene mappato in un numero intero (che è espresso da sequenza di bit) utilizzando dei codici

Il codice più usato è l'*ASCII (American Standard Code for Information Interchange)* a 8 bit che contiene:

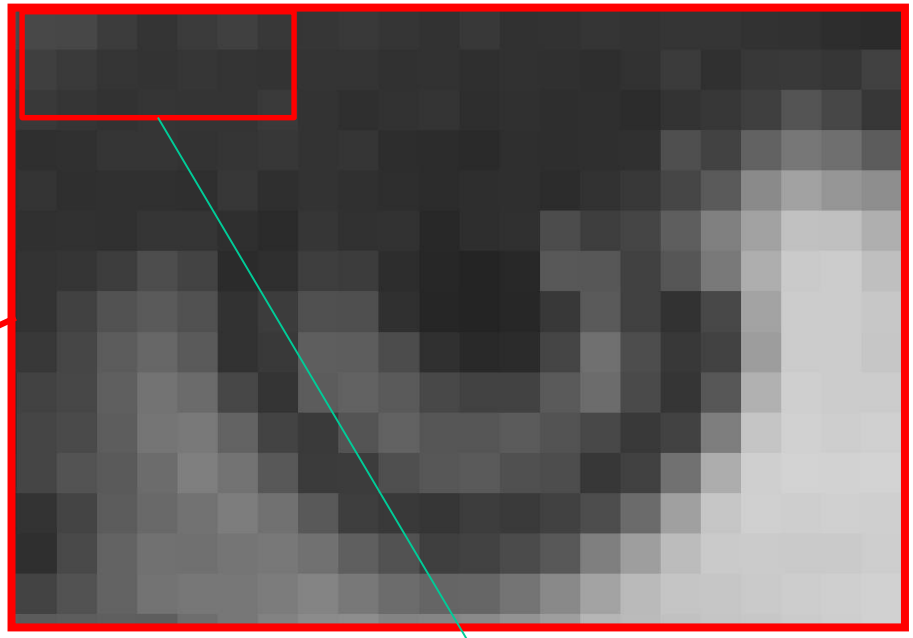
- Caratteri alfanumerici
- Caratteri simbolici (es. punteggiatura, @&%$ etc..)
- Caratteri di comando (es. termina riga, vai a capo, tab)

La codifica ASCII (parziale)

DEC	CAR	DEC	CAR	DEC	CAR	DEC	CAR	DEC	CAR
48	0	65	A	75	K	97	a	107	k
49	1	66	B	76	L	98	b	108	l
50	2	67	C	77	M	99	c	109	m
51	3	68	D	78	N	100	d	110	n
52	4	69	E	79	O	101	e	111	o
53	5	70	F	80	P	102	f	112	p
54	6	71	G	81	Q	103	g	113	q
55	7	72	H	82	R	104	h	114	r
56	8	73	I	83	S	105	i	115	s
57	9	74	J	84	T	106	j	116	t
				85	U			117	u
				86	V			118	v
				87	W			119	w
				88	X			120	x
				89	Y			121	y
				90	Z			122	z

Codifica delle Immagini

Le immagini nei calcolatori sono digitali, i.e. tabella di pixel, ciascuno caratterizzato da uno o più valori di intensità.



123	122	134	121	132	133	145	134
122	121	125	132	124	121	116	126
119	127	137	119	139	127	128	131

Codifica delle Immagini

Le immagini nei calcolatori sono digitali, i.e. tabella di pixel, ciascuno caratterizzato da uno o più valori di intensità.

Canale rosso

Canale verde

Canale blu

Codifica delle Immagini

- Esistono diverse codifiche dell'immagine, non sempre questa viene scritta pixel per pixel.
- È spesso conveniente rappresentare l'immagine secondo codifiche che permettano di ridurre le dimensioni.
- Codifiche *lossless*: permettono, senza perdita di informazione, di comprimere l'immagine
  - e.g, non serve ripetere il valore nelle aree costanti, conviene registrare le variazioni (e.g. gif, png)
- Codifiche *lossy:* perdita di informazione e di qualità
  - e.g., le immagini jpeg sono compresse a blocchi, ogni blocco contiene meno dettagli dell'immagine originale.

Sistemi Informatici

- Hardware
- Software

Cos'è l'Informatica?

Scienza della rappresentazione e dell'elaborazione dell'informazione.

- ***Scienza:*** ovvero una **conoscenza sistematica** e **rigorosa** di tecniche e metodi.
- ***Informazione:*** l'oggetto dell'investigazione scientifica (informazione intesa come entità astratta e come tecnologie per la sua gestione)
- ***Rappresentazione:*** il modo in cui l'informazione viene strutturata e trasformata in dati fruibili da macchine
- ***Elaborazione:*** uso e trasformazione dell'informazione per un dato scopo. L'elaborazione deve poter essere eseguita da **macchine** che processano dati.

[da «Informatica Arte e Mestiere»]

Sistema Informatico

- Sistemi informatico: L'esecutore dei programmi
- PC, laptop, telefoni, smartphones, server con migliaia di utenti etc… tutti sistemi informatici
- Sono oggetti complessi, costruiti da diverse parti che interagiscono tra loro.
- I sistemi informatici sono composti da
  - **Hardware**: i componenti fisici del sistema
  - **Software**: i programmi eseguiti dal sistema (es. web browser, mail, suite office, sistema operativo, compilatori, IDE… i programmi che scriveremo)
- Consideriamo inizialmente l'hardware dei sistemi informatici.

La Macchina di Von Neumann

Un modello dell'architettura dei calcolatori

La Macchina di Von Neumann

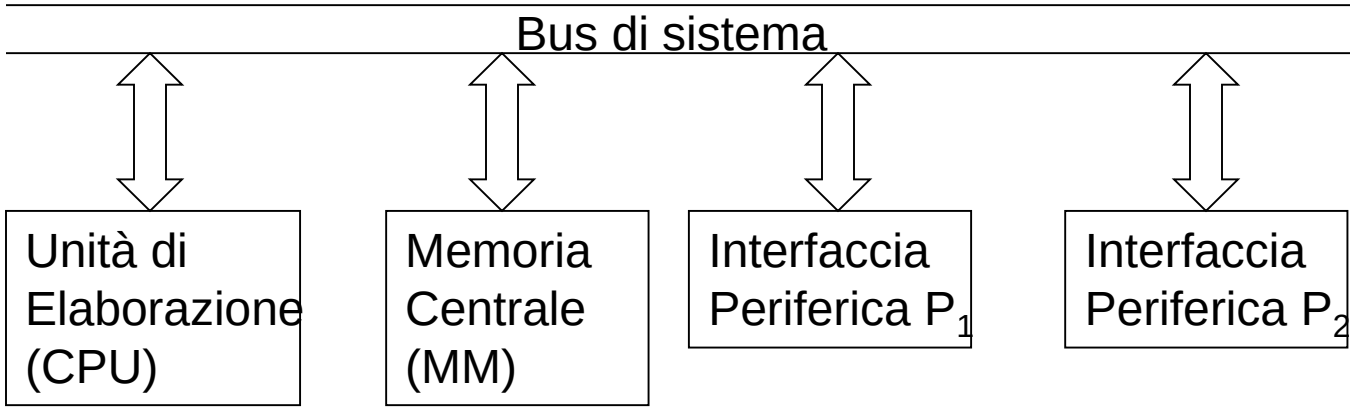
Modello composto da **quattro elementi funzionali**

- **Unità di elaborazione (CPU):** interpretazione ed esecuzione dei programmi, coordinamento macchina
- **Memoria centrale:** contiene dati ed istruzioni
- **Interfacce delle periferiche:** scambio informazioni con mondo esterno (e.g, stampante, tastiera, mouse, rete, schermo, HDD ..)
- **Bus di sistema:** collega gli altri elementi funzionali

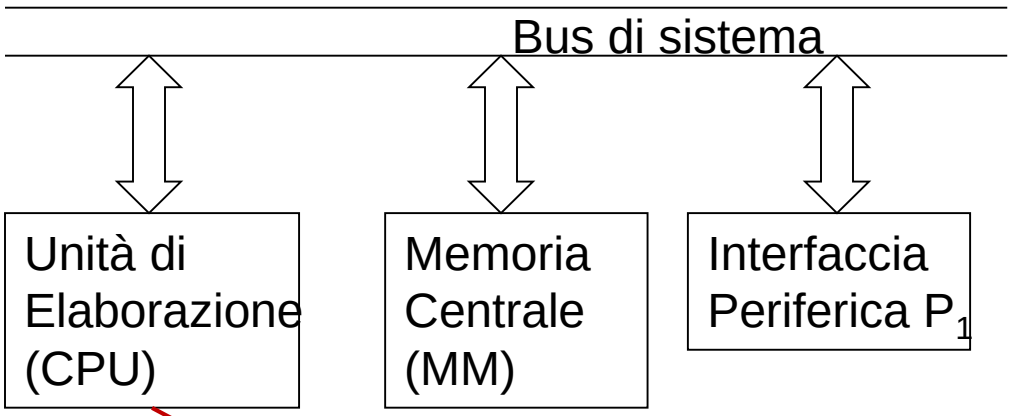
La Macchina di Von Neumann

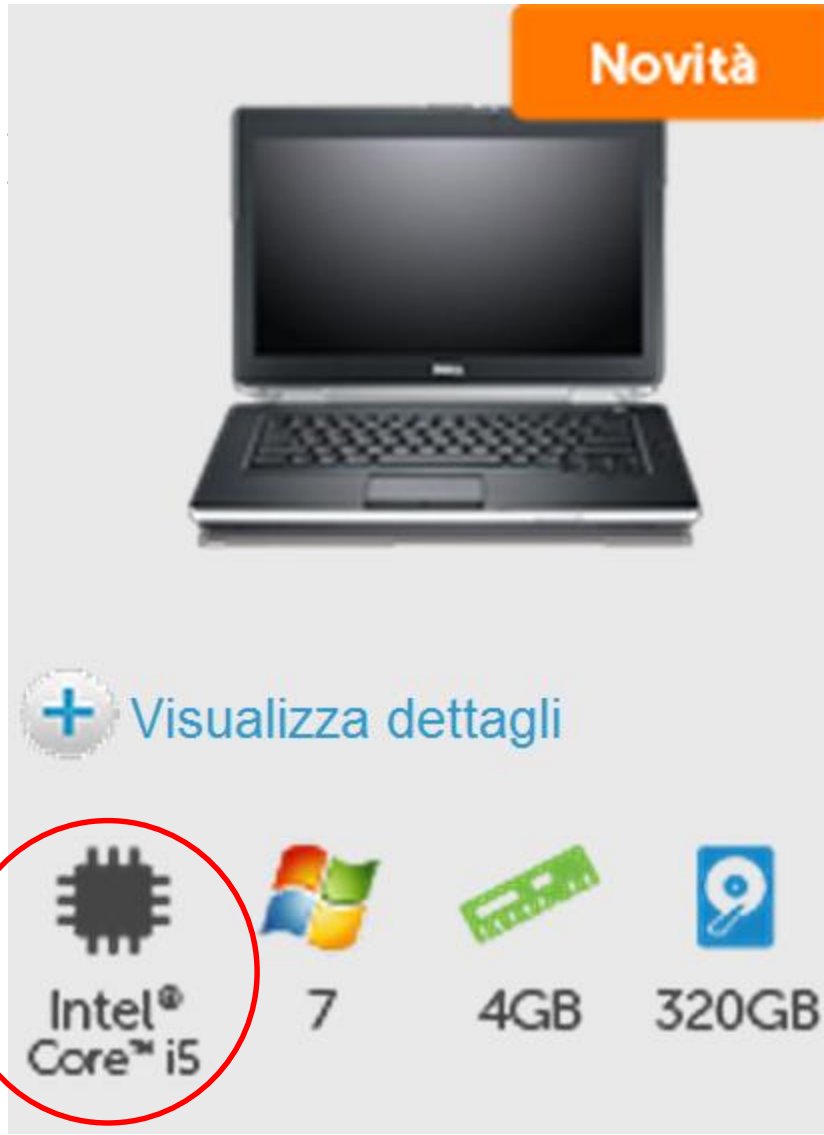
Modello composto da **quattro elementi funzionali**

- **Unità di elaborazione (CPU):** interpretazione ed esecuzione dei programmi, coordinamento macchina
- **Memoria centrale:** contiene dati ed istruzioni
- **Interfacce delle periferiche:** scambio informazioni con mondo esterno (e.g, stampante, tastiera, mouse, rete, schermo, HDD ..)
- **Bus di sistema:** collega gli altri elementi funzionali

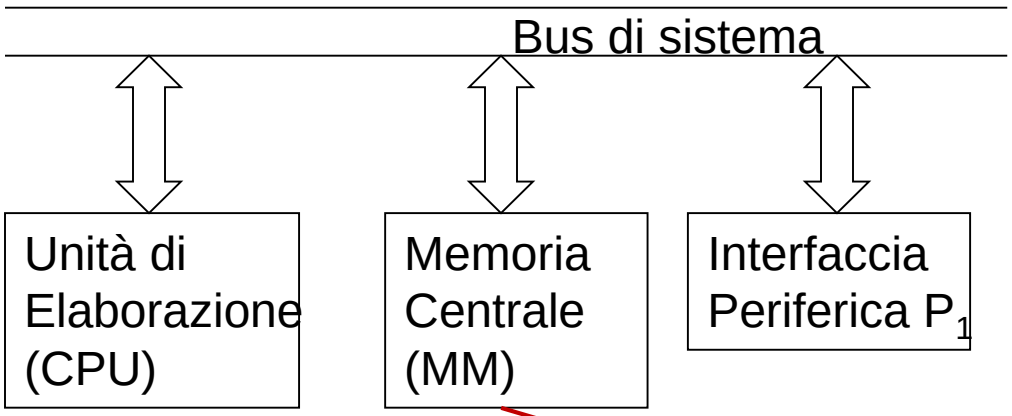


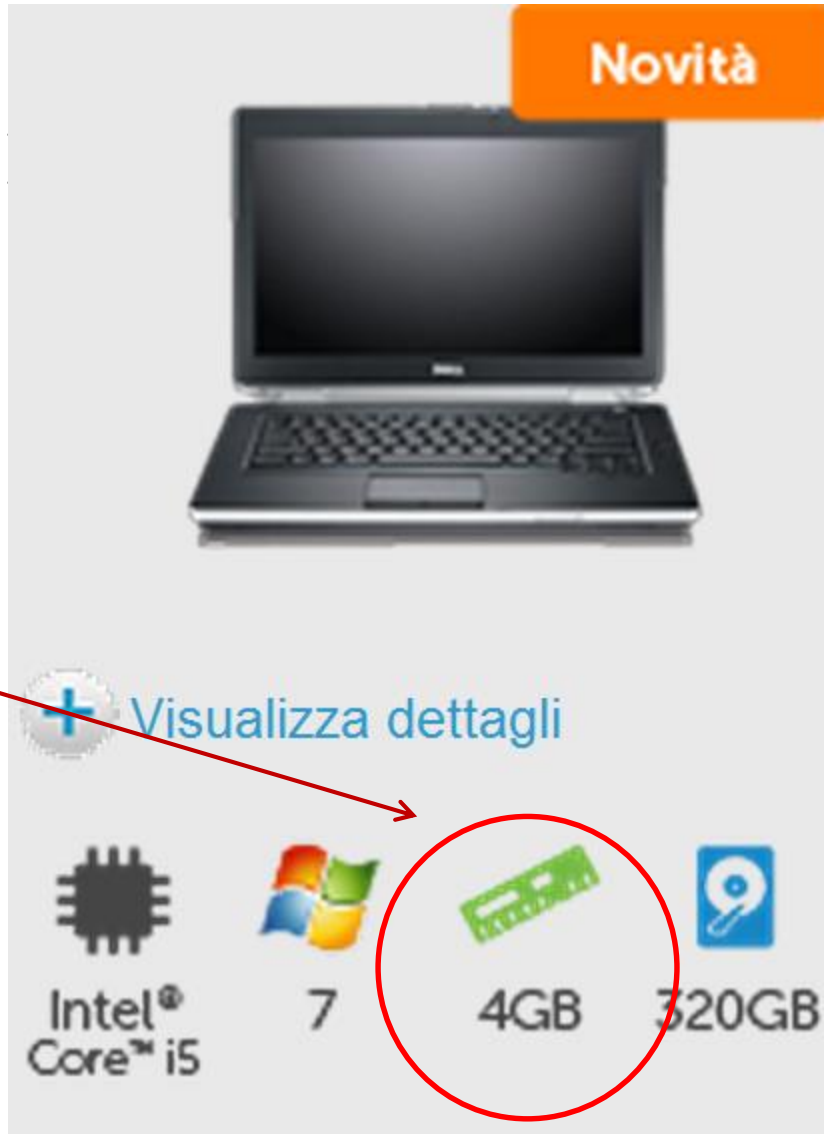
La Macchina di Von Neumann



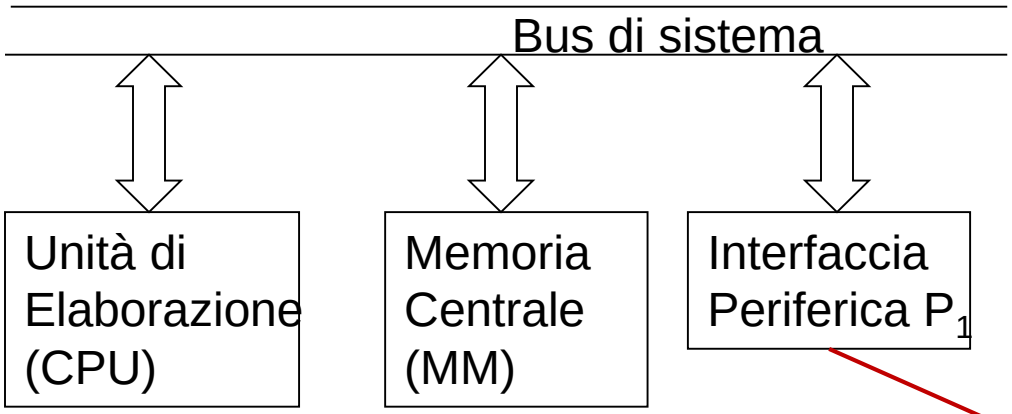


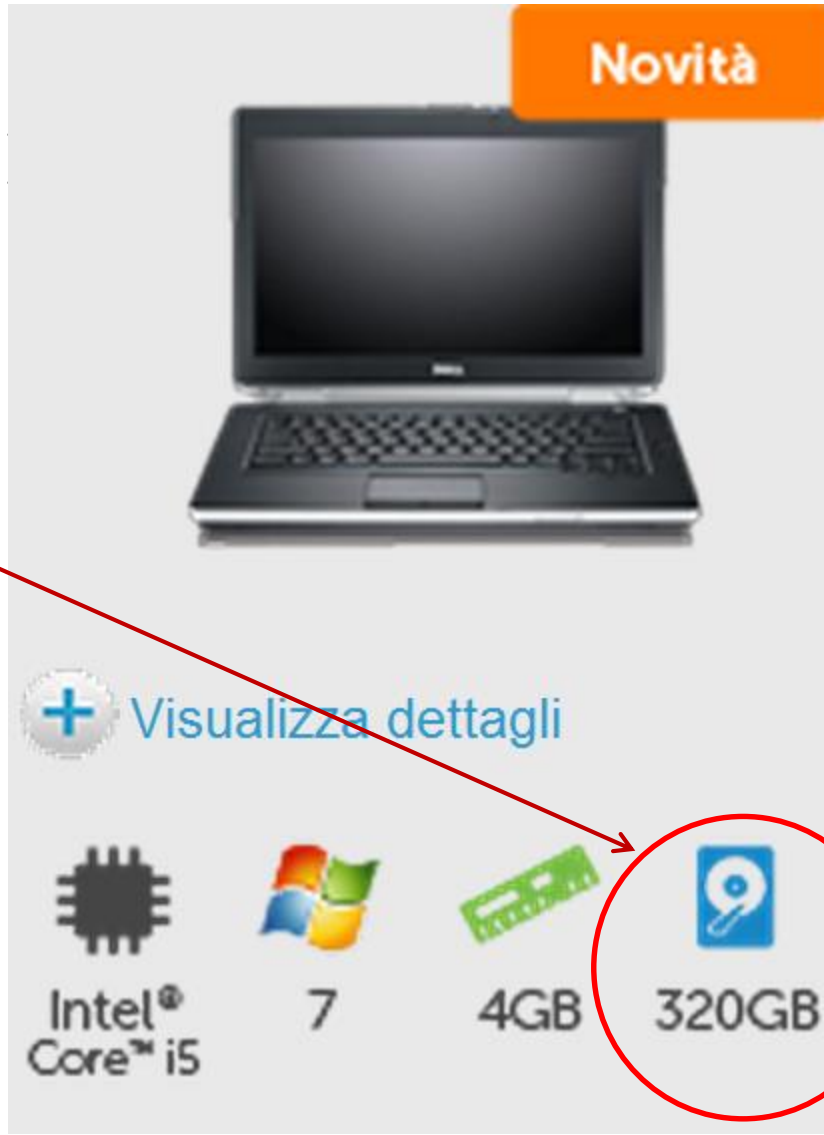
La Macchina di Von Neumann



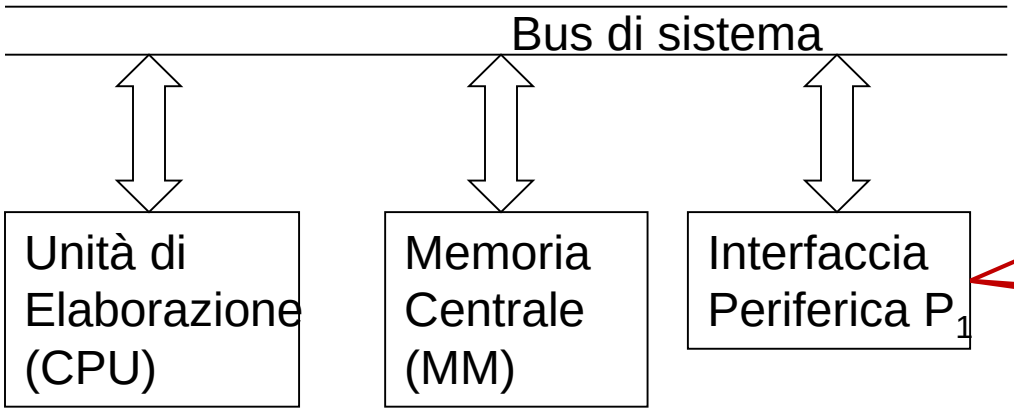


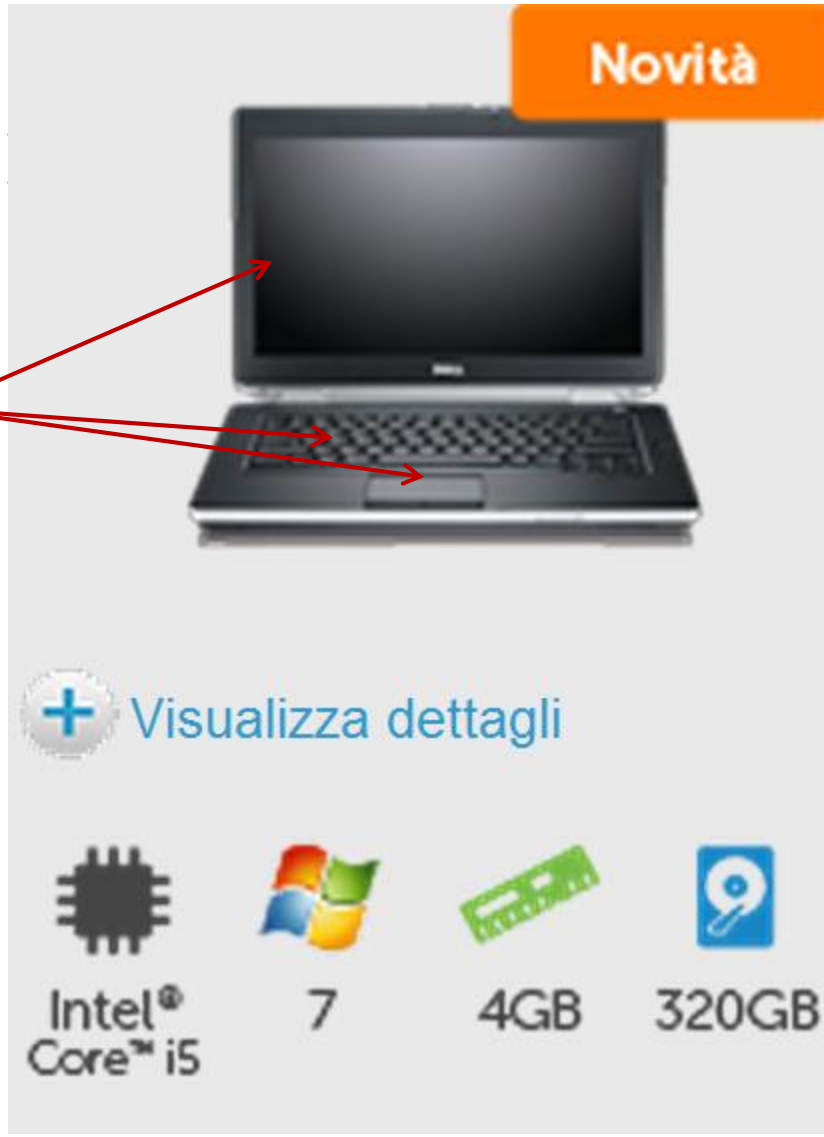
La Macchina di Von Neumann





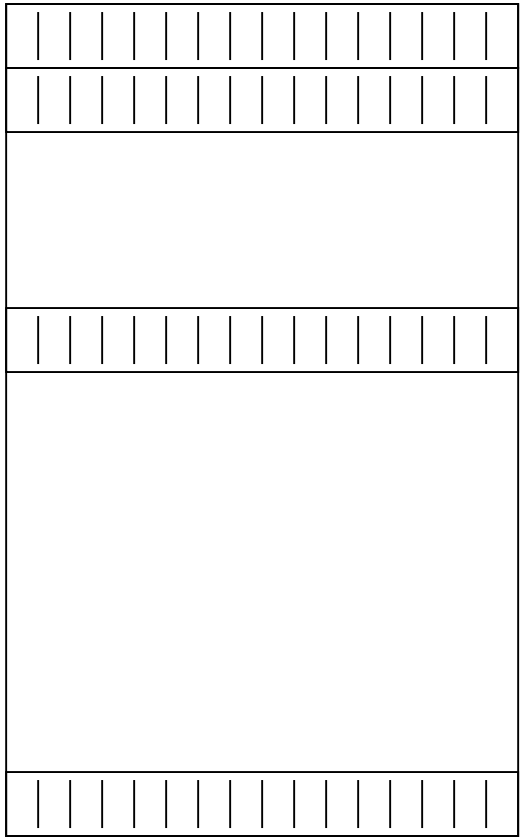
La Macchina di Von Neumann





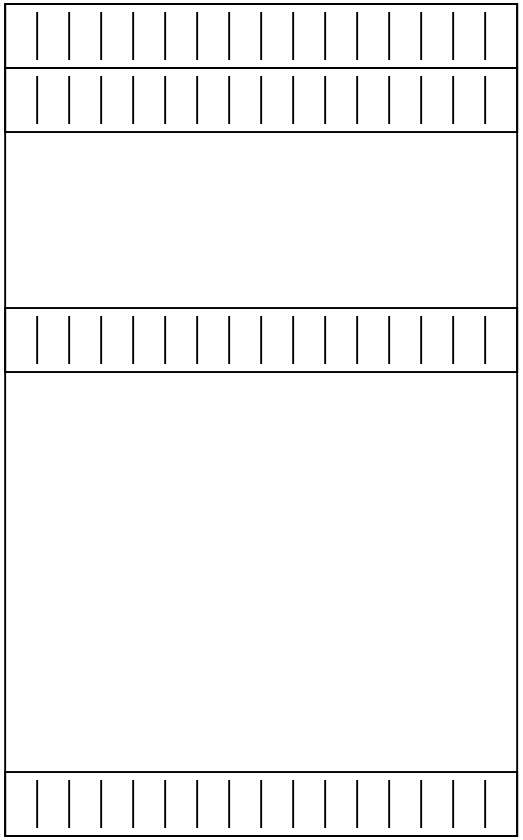
La Memoria Centrale (MM)

- Contiene i **programmi (sequenza di istruzioni)** in **esecuzione** ed i relativi **dati**



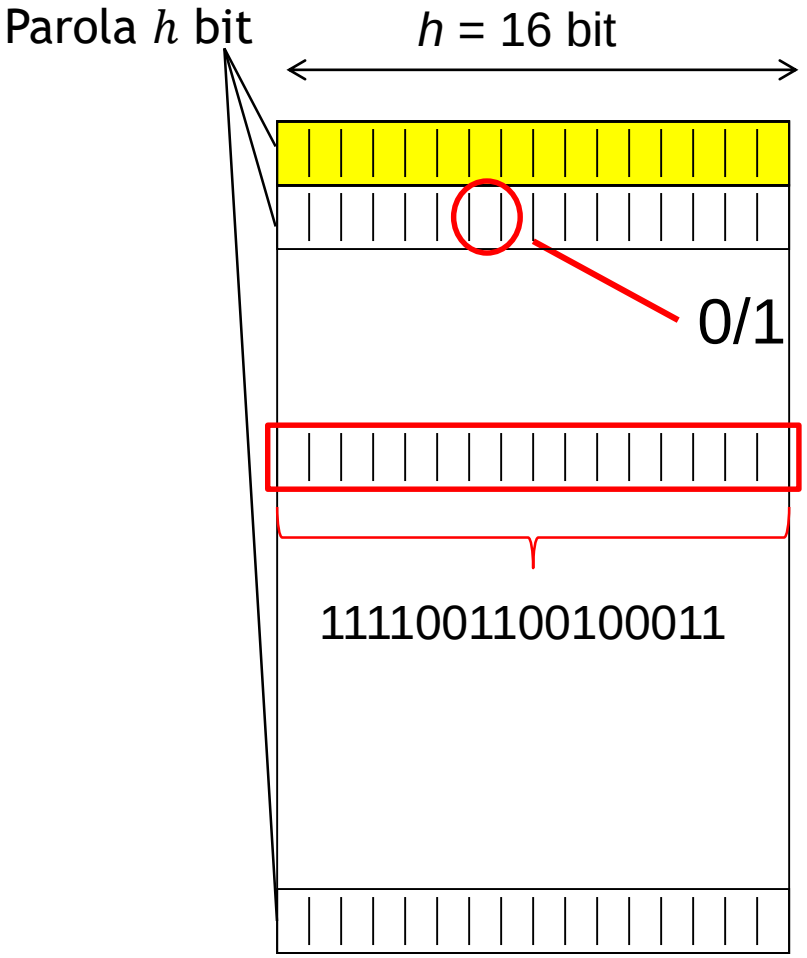
La Memoria Centrale (MM)

- Contiene i **programmi (sequenza di istruzioni)** in **esecuzione** ed i relativi **dati**
- È schematizzata come una sequenza di celle.



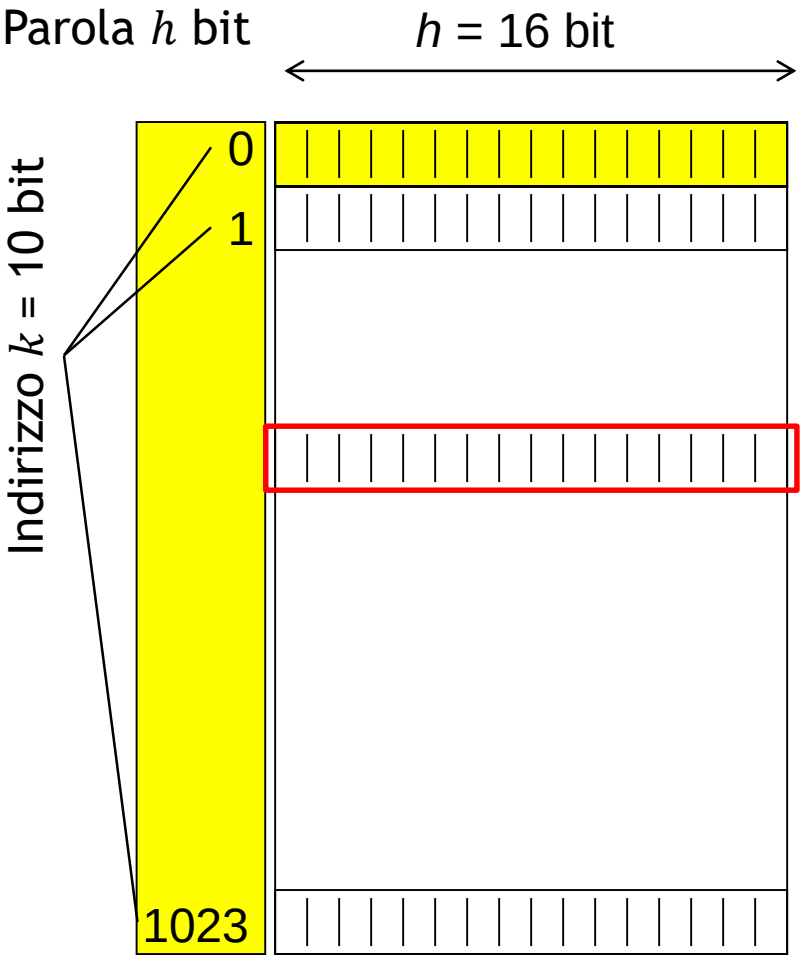
La Memoria Centrale (MM)

- Contiene i **programmi (sequenza di istruzioni)** in **esecuzione** ed i relativi **dati**
- È schematizzata come una sequenza di celle.
- Ogni cella contiene *h* bit, i.e., una Parola (*word*)



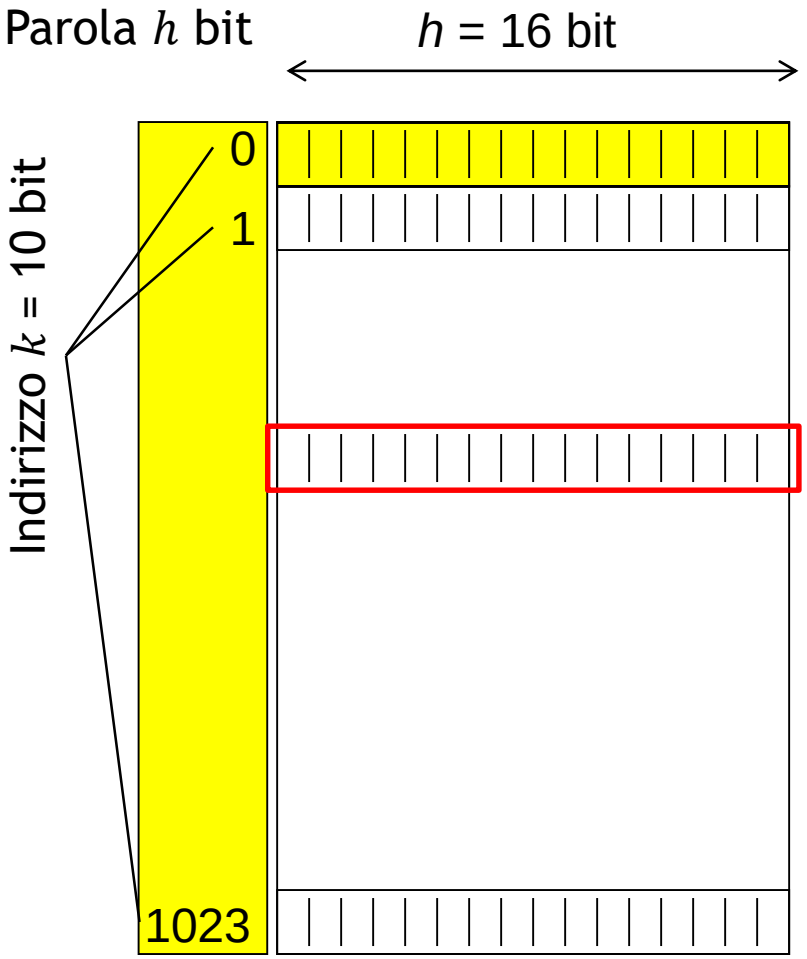
La Memoria Centrale (MM)

- Contiene i **programmi (sequenza di istruzioni)** in **esecuzione** ed i relativi **dati**
- È schematizzata come una sequenza di celle.
- Ogni cella contiene *h* bit, i.e., una Parola (*word*)
- Ogni cella ha un indirizzo
- Se ho a disposizione *k* bit per scrivere l'indirizzo, lo spazio di indirizzamento è 2^k celle



La Memoria Centrale (MM)

- Contiene i **programmi (sequenza di istruzioni)** in **esecuzione** ed i relativi **dati**
- È schematizzata come una sequenza di celle.
- Ogni cella contiene *h* bit, i.e., una Parola (*word*)
- Ogni cella ha un indirizzo
- Se ho a disposizione *k* bit per scrivere l'indirizzo, lo spazio di indirizzamento è 2^k celle



Avere più celle dello spazio di indirizzamento è come un palazzo senza scale che ha meno pulsanti dell'ascensore dei piani

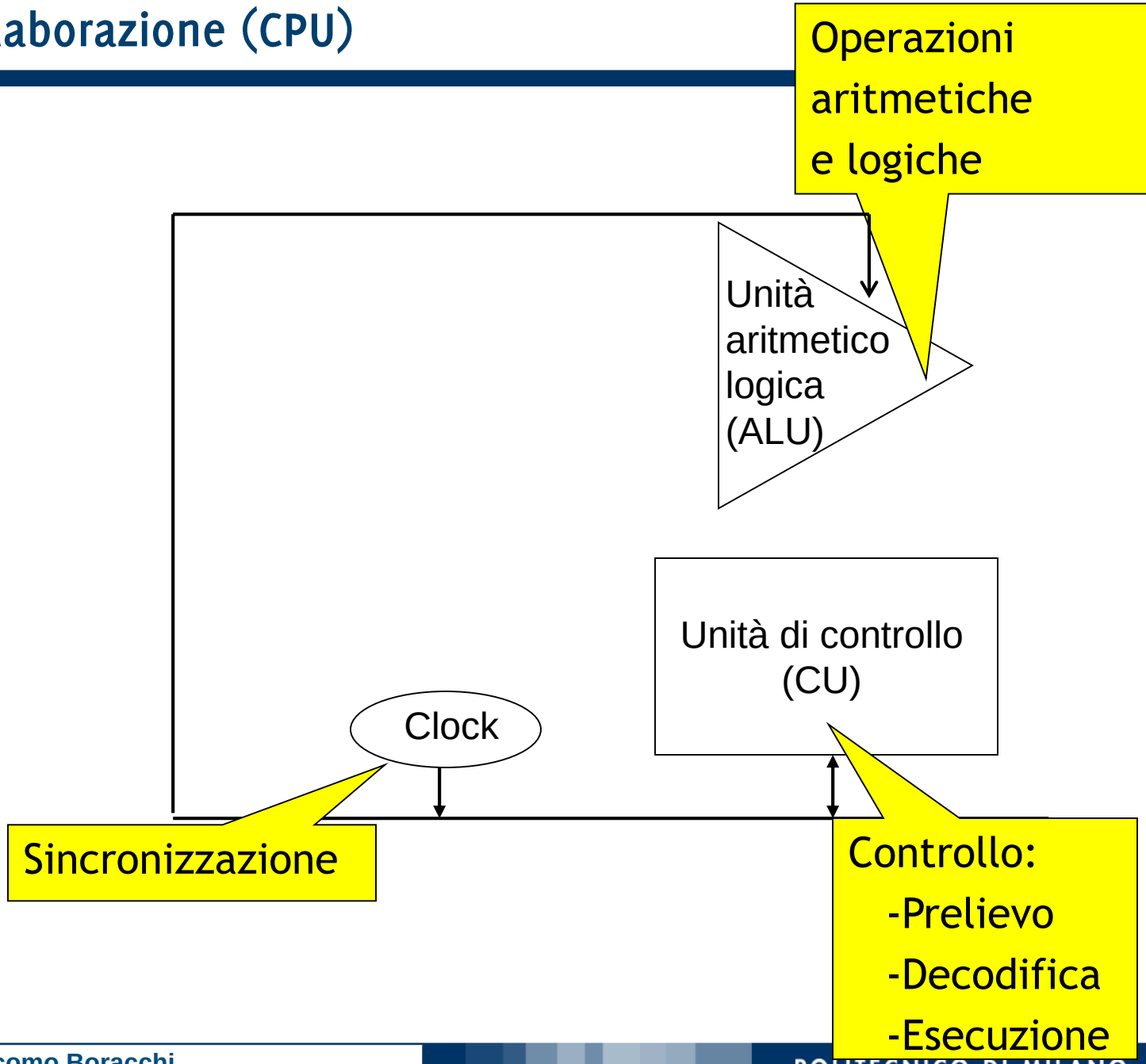
La Memoria Centrale (MM)

- La MM contiene i **programmi in esecuzione**: ogni **dato e ogni istruzione**, prima di essere elaborato, viene copiato in memoria centrale.
- Diversi tipi di Memorie
  - RAM (*Random Access Memory*) memoria volatile.
  - ROM *(Read Only Memory)* memoria permanente.
  - EPROM *(Erasable Programmable ROM)* riprogrammabile
- **L'HDD** è memoria permanente ma **non è memoria centrale** ed in riferimento alla macchina di Von Neumann è una periferica.

L'Unità di Elaborazione (CPU)

- La *Central Processing Unit* (CPU) **coordina** il funzionamento del **calcolatore** ed **esegue** i **programmi**:
  estrae, decodifica ed esegue le istruzioni in memoria.
- Le istruzioni possono comportare **elaborazione** o **trasferimento** dell'informazione
- La CPU contiene a sua volta:
  - l'**Unità di Controllo** che preleva e decodifica istruzioni dalla MM, invia segnali per eseguire le istruzioni
  - Il **Clock di sistema**, opera come un metronomo per la CPU
  - L'**Unità Aritmetico Logica:** operazioni aritmetiche e logiche

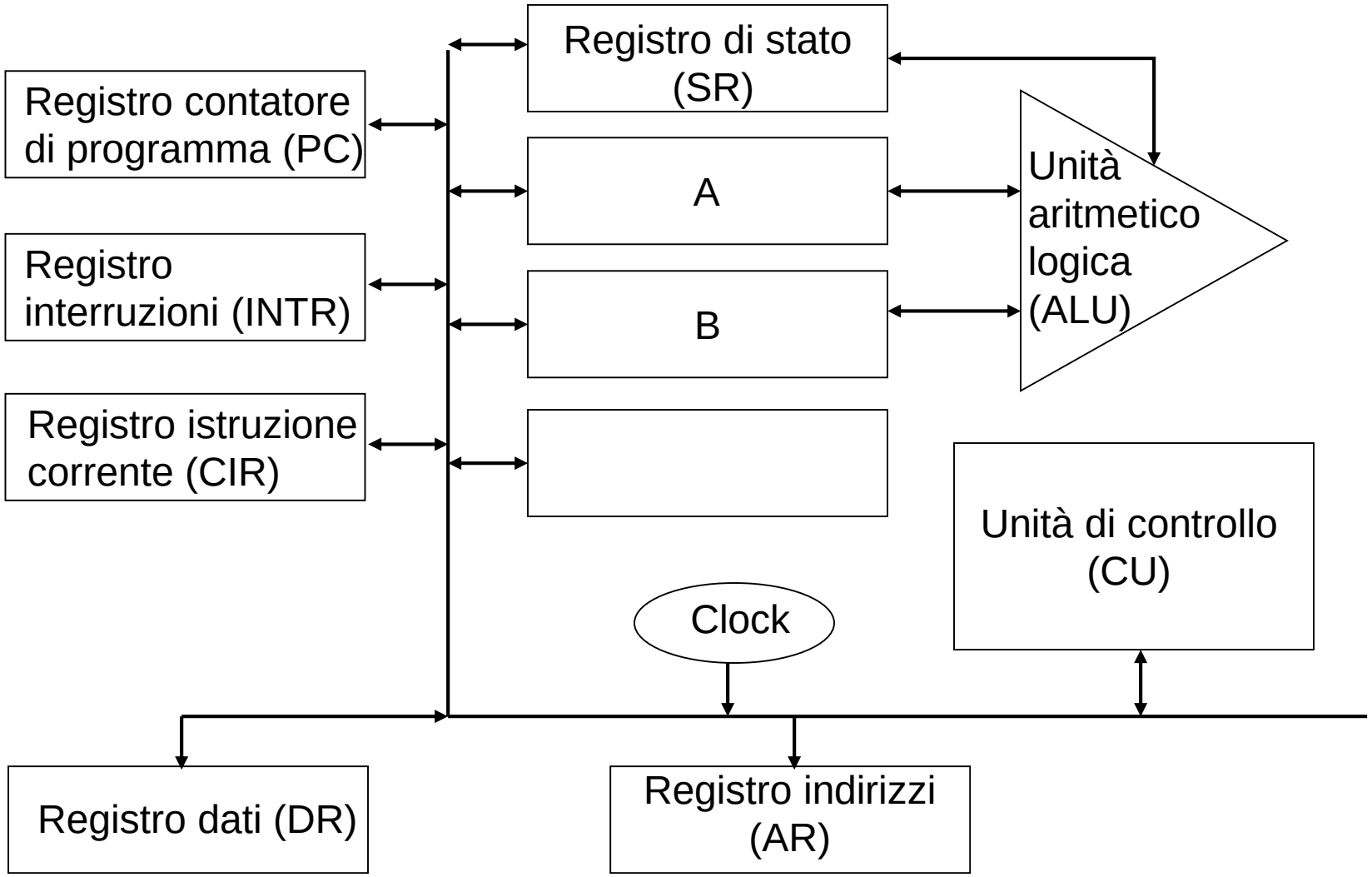
L'Unità di Elaborazione (CPU)



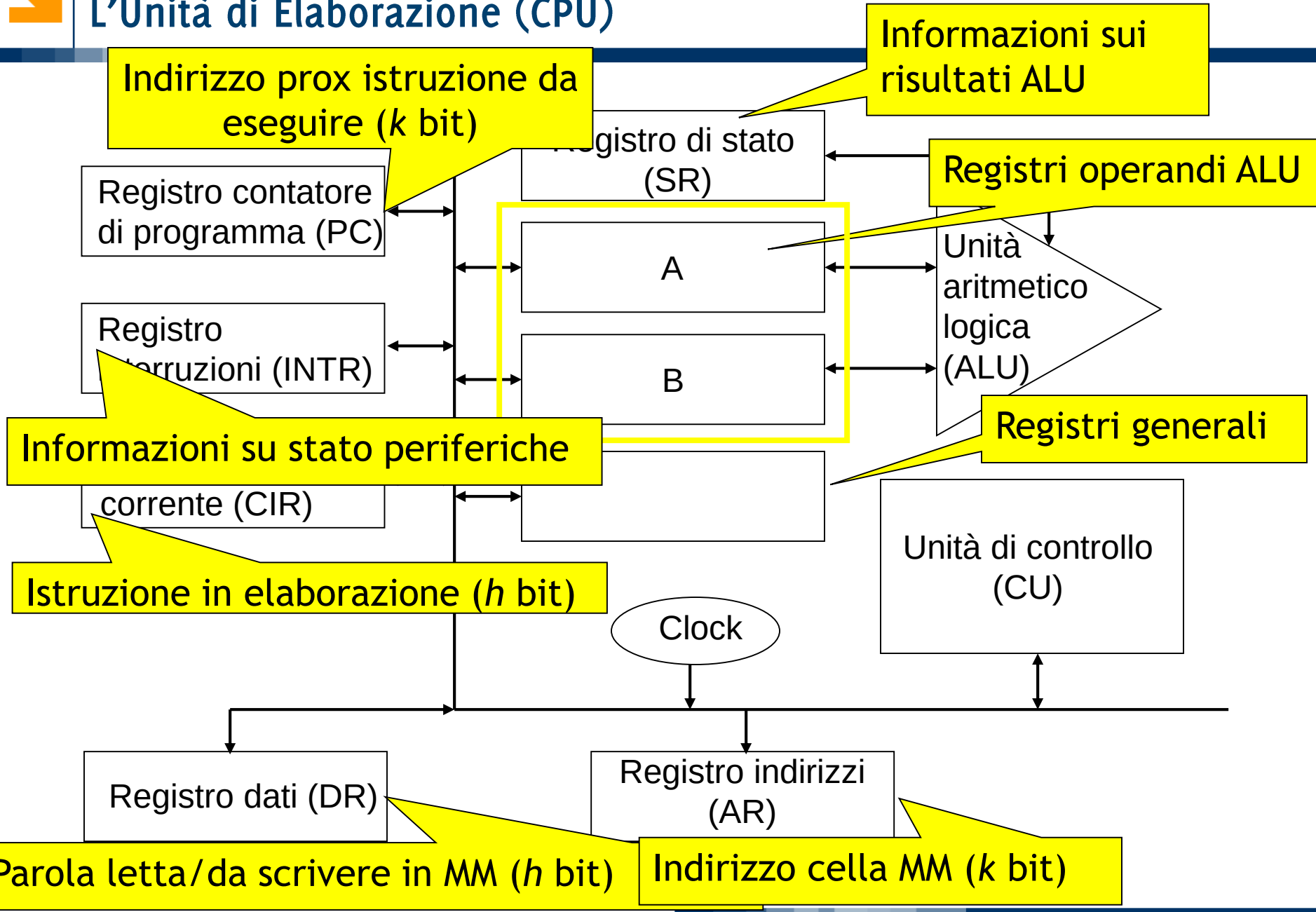
L'Unità di Elaborazione (CPU)

- La *Central Processing Unit* (CPU) **coordina** il funzionamento del **calcolatore** ed **esegue** i **programmi**:
  estrae, decodifica ed esegue le istruzioni in memoria.
- Le istruzioni possono comportare **elaborazione** o **trasferimento** dell'informazione
- La CPU contiene a sua volta:
  - l'**Unità di Controllo** che preleva e decodifica istruzioni dalla MM, invia segnali per eseguire le istruzioni
  - Il **Clock di sistema**, opera come un metronomo per la CPU
  - L'**Unità Aritmetico Logica:** operazioni aritmetiche e logiche
- La CPU contiene inoltre molti **registri**: memorie «rapide» per informazioni richieste dalla CU (es due numeri da sommare, il loro risultato)

L'Unità di Elaborazione (CPU)



L'Unità di Elaborazione (CPU)

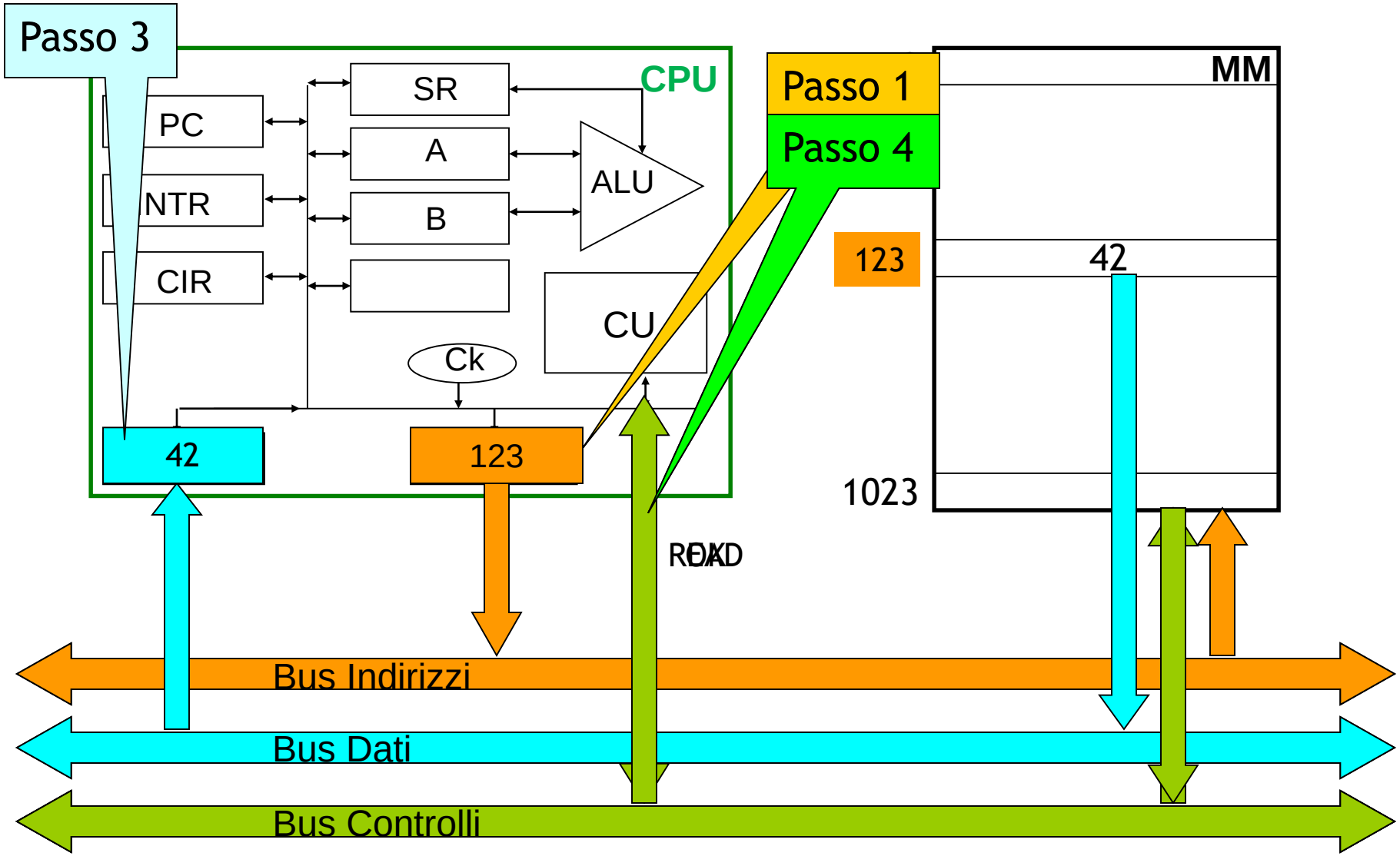


Bus di Sistema

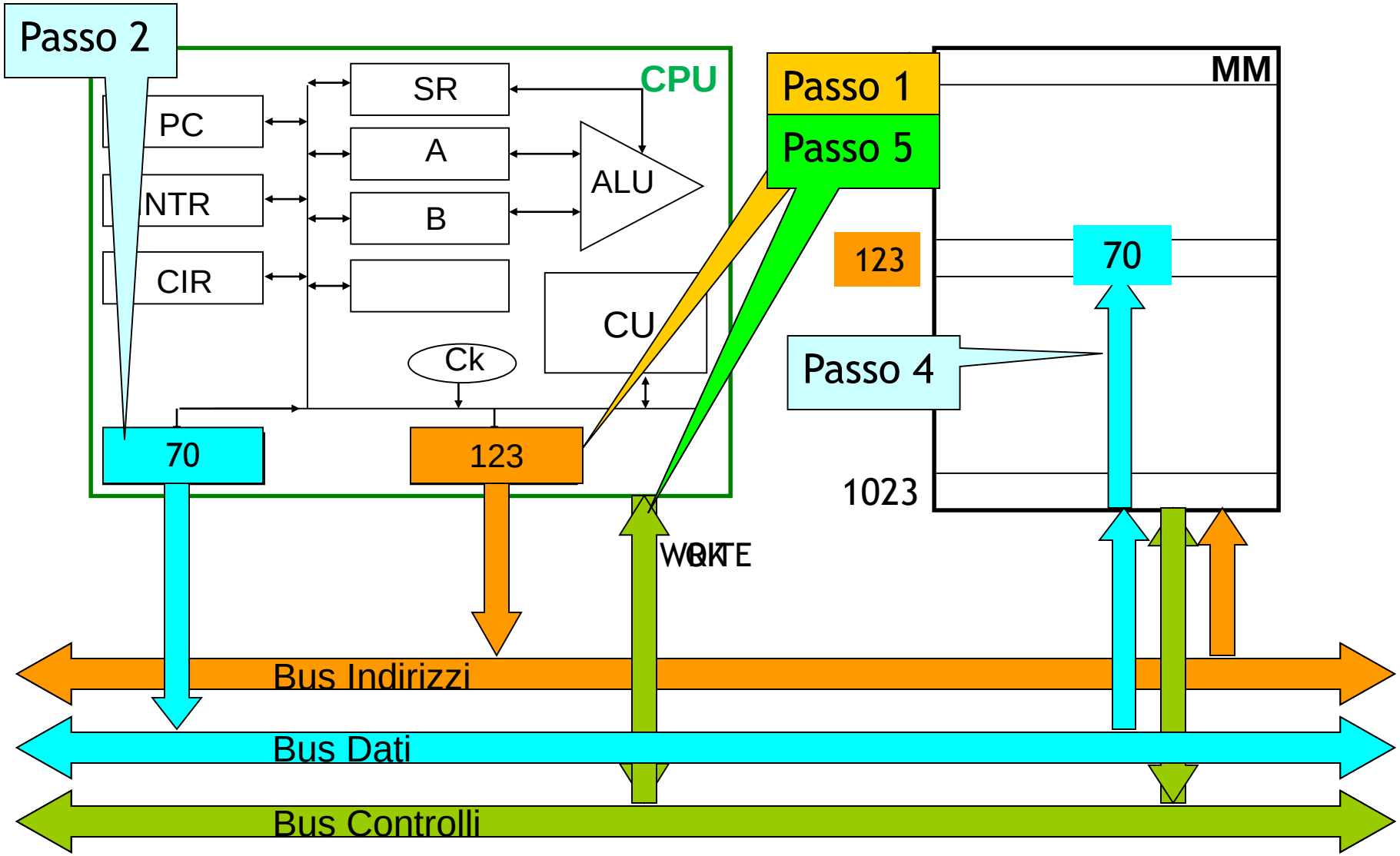
- È un insieme di connessioni che permettono di trasferire l'informazione tra **due** entità funzionali
  - Un'entità trasmette, l'altra riceve
- Due soli tipi di connessioni logiche, **stabilite** dalla **CPU**:
  - CPU (**master**) - memoria (**slave**)
  - CPU (**master**) - interfaccia periferica (**slave**)

  le connessioni fisiche sono sempre presenti.
- Ci sono **tre tipi di linee**, con tre funzionalità diverse
  - Bus **dati**
  - Bus **indirizzi**
  - Bus **controlli**: usato dal master per **trasmettere le istruzioni** da eseguire allo slave, e dallo slave per dare **feedback**

Sequenza di Lettura



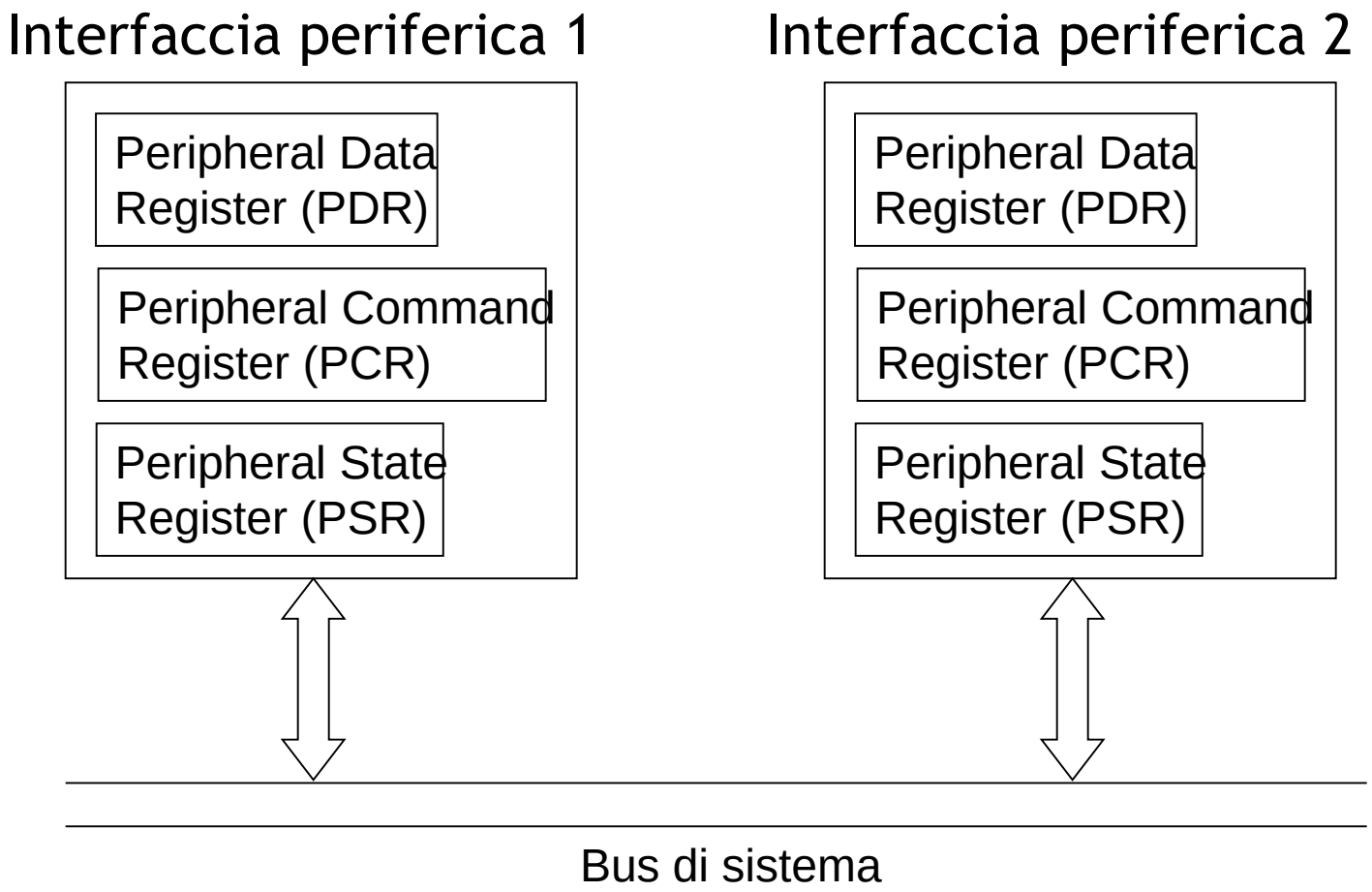
Sequenza di Scrittura



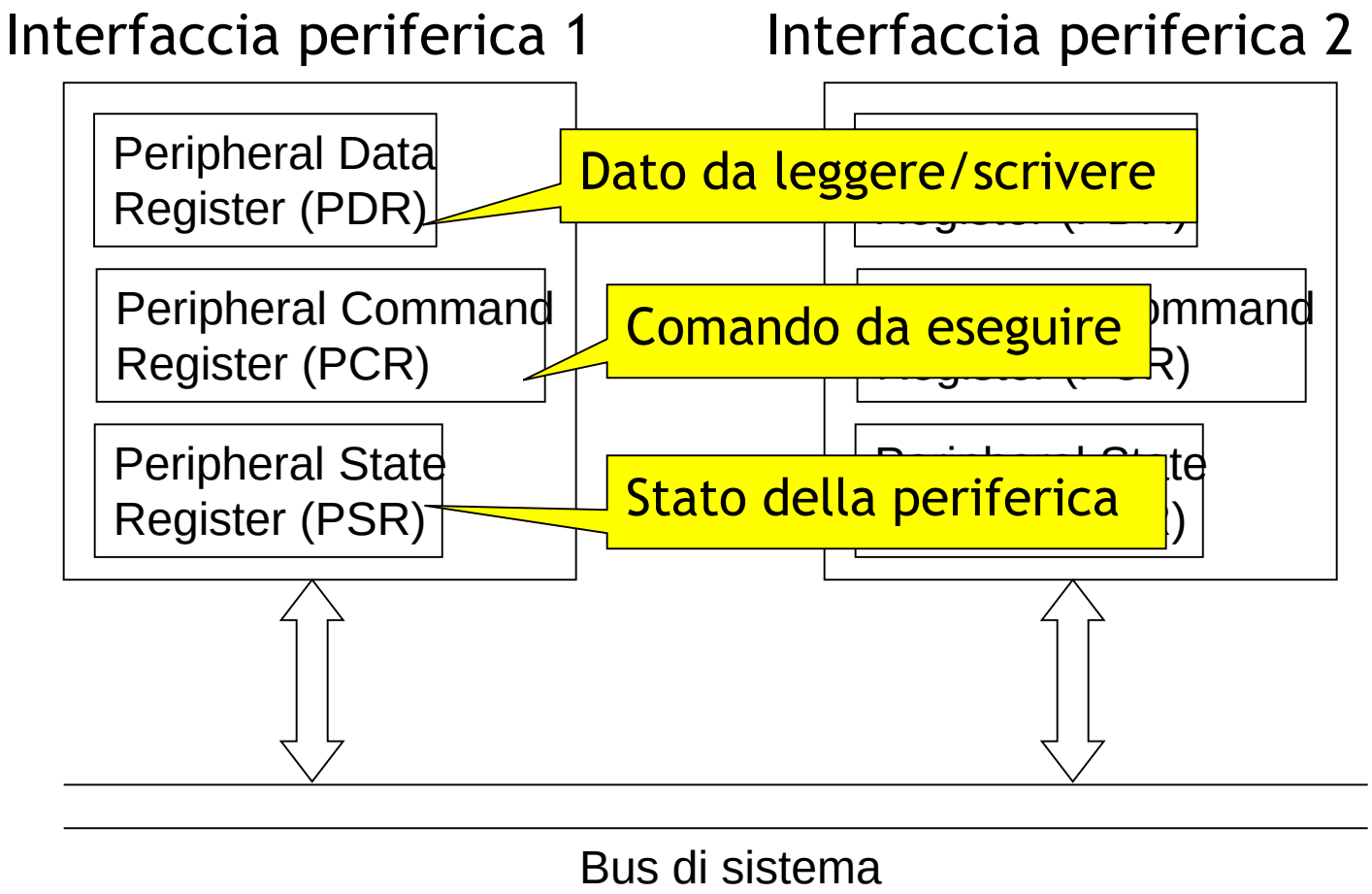
Interfacce alle Periferiche

- Le interfacce collegano il calcolatore a periferiche esterne
- Ogni interfaccia contiene dei registri per lo scambio dei dati con la periferica
  - Registro dati
  - Registro comando della periferica
  - Registro di stato

Interfacce alle Periferiche



Interfacce alle Periferiche



I Programmi Nella Macchina di Von Neumann

I Programmi nella Macchina di Von Neumann

Le **istruzioni** sono (necessariamente) codificate in **binario** e, **come i dati,** sono salvate in **parole nella MM**

I Programmi nella Macchina di Von Neumann

Le **istruzioni** sono (necessariamente) codificate in **binario** e, **come i dati,** sono salvate in **parole nella MM**

Supponiamo una MM con parole da $h = 16$ bit ed indirizzi da $k = 10$ bit, con istruzioni così codificate:

Codice operativo (4bit) oo Indirizzo Operando (10bit)

ad esempio, 0100000000010000

I Programmi nella Macchina di Von Neumann

Le **istruzioni** sono (necessariamente) codificate in **binario** e, **come i dati,** sono salvate in **parole nella MM**

Supponiamo una MM con parole da $h = 16$ bit ed indirizzi da $k = 10$ bit, con istruzioni così codificate:

Codice operativo (4bit) oo Indirizzo Operando (10bit)

ad esempio, 0100000000010000

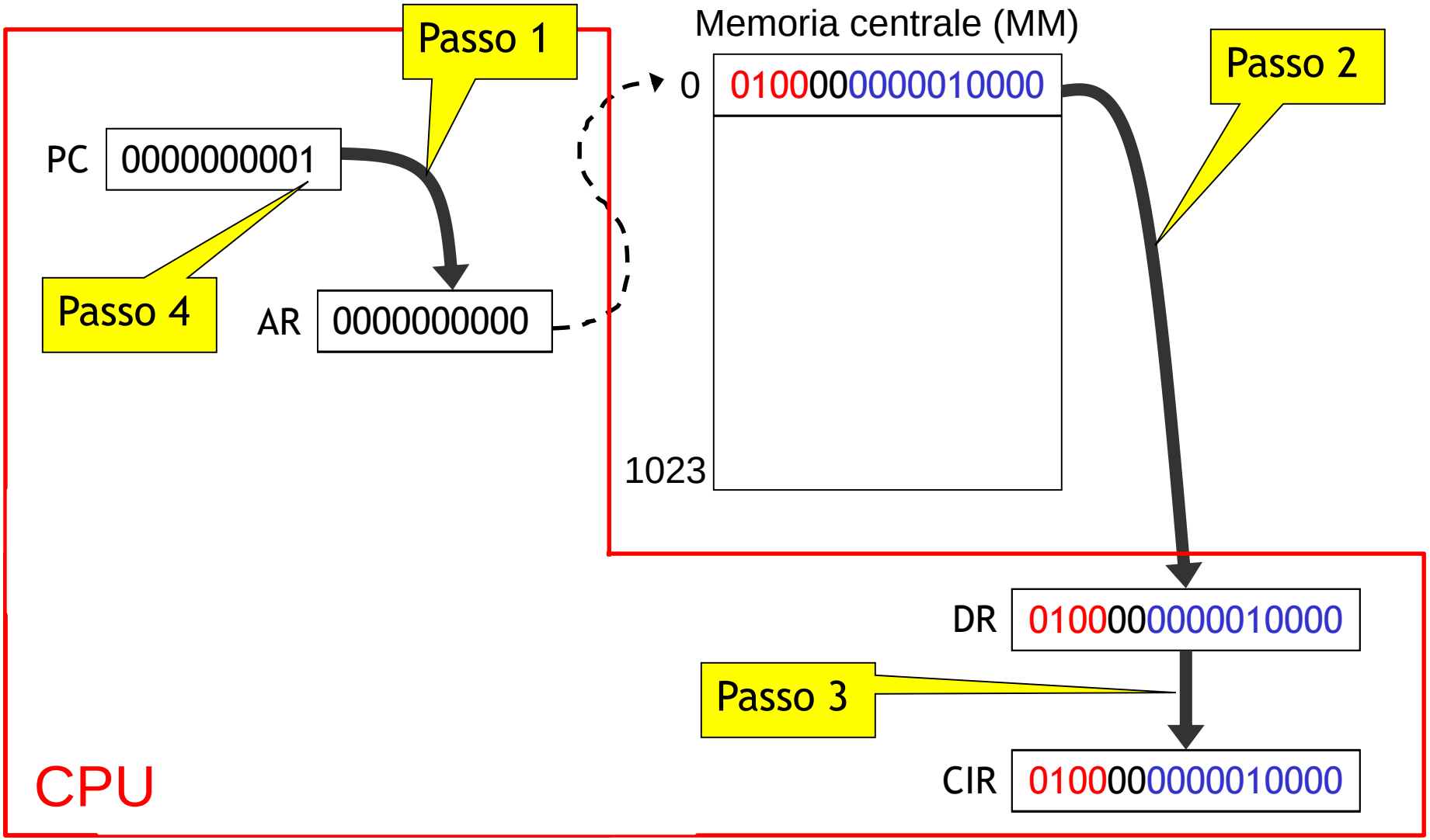
Consideriamo le seguenti istruzioni eseguibili dalla CPU

- Lettura da periferica, scrittura su periferica
- Caricare un dato da MM in un registro della CPU (*load*)
- Salvare in MM un dato di un registro della CPU (*store*)
- Operazioni aritmetiche (le gestisce la ALU)
- Istruzioni di salto (per cambiare il flusso di esecuzione)

Le Tre Fasi Per Eseguire un'Istruzioni

1. **Fetch:** Acquisizione dell'istruzione dalla MM
   1. Trasferimento da PC a AR dell' indirizzo cella contenente l'istruzione da eseguire.
   2. Lettura dalla MM della cella all'indirizzo in AR, contenuto trasferito sul DR (l'istruzione è un dato)
   3. Sposta da DR a CIR (riferimento istr. in esecuzione)
   4. Incrementa PC (definisce la prossima istruzione: incremento di 1 = sequenzialità)
2. **Decodifica**: riguarda il codice operativo, legge dal CIR
3. **Esecuzione**: dipende dall'istruzione specifica.

Fase di Fetch

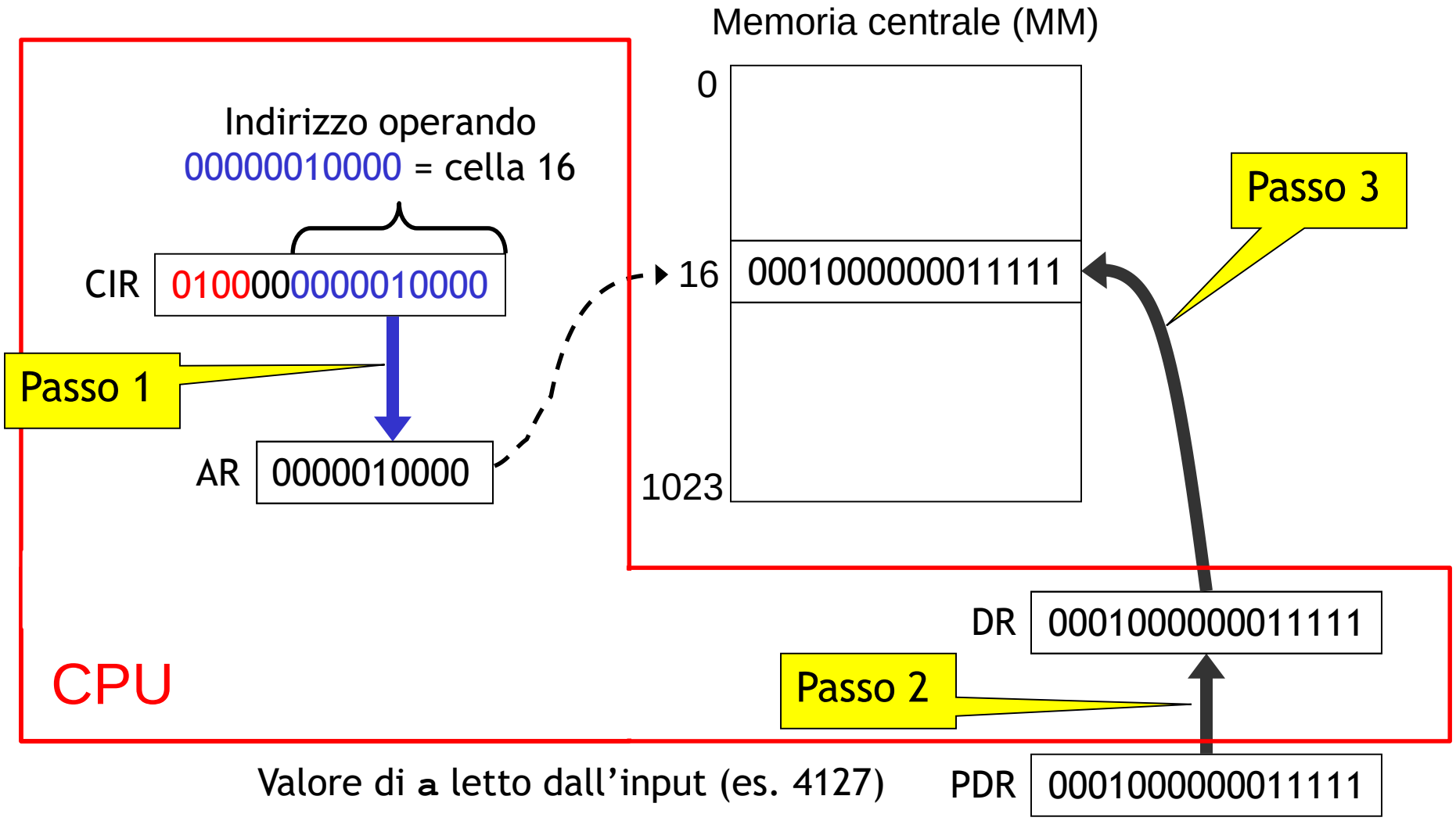


Fase di Decodifica 1ª istruzione

CIR 0100000000010000

Codice operativo 0100 = leggi da input

Fase di esecuzione: esempio lettura da input



Esempio

- Si consideri una Macchina di Von Neumann con Parola a 16 bit, indirizzi a 10 bit e codice operativo 4 bit
- Vogliamo calcolare il valore dell'espressione:

  $$(a+b) \cdot (c+d)$$

  leggendo i valori delle variabili **a**, **b**, **c**, **d** dal dispositivo di ingresso e scrivendo il risultato della valutazione sul dispositivo di uscita.

Esempio

1. Leggi dal dispositivo di ingresso il valore delle variabili **`a`**, **`b`**, **`c`**, **`d`**
2. Somma il valore di **`a`** al valore di **`b`**
3. Salva il risultato parziale ottenuto
4. Somma il valore di **`c`** al valore di **`d`**
5. Moltiplica il risultato parziale appena ottenuto con quello precedentemente salvato
6. Scrivi sul dispositivo di uscita il risultato della valutazione complessiva
7. Termina l'esecuzione del programma.

Esempio

1. Leggi dal dispositivo di ingresso il valore delle variabili **`a`**, **`b`**, **`c`**, **`d`**
2. Somma il valore di **`a`** al valore di **`b`**
3. Salva il risultato parziale ottenuto
4. Somma il valore di **`c`** al valore di **`d`**
5. Moltiplica il risultato parziale appena ottenuto con quello precedentemente salvato
6. Scrivi sul dispositivo di uscita il risultato della valutazione complessiva
7. Termina l'esecuzione del programma.

Il programma deve essere tradotto in opportuni codici operativi e indirizzi di celle di memoria!

Esempio: più nel dettaglio

1. Leggi dal dispositivo di ingresso il valore delle variabili **a**

1. Scrivi nella cella di memoria centrale riservata al valore di **a** il valore letto dal registro dati della periferica.
   - Occorre la posizione di **a,b,c,d**

Esempio: più nel dettaglio

1. Leggi dal dispositivo di ingresso il valore delle variabili **a**
2. Somma il valore di **a** al valore di **b**

1. Scrivi nella cella di memoria centrale riservata al valore di **a** il valore letto dal registro dati della periferica.
   - Occorre la posizione di **a,b,c,d**
2. Somma il valore di **a** al valore di **b**

   2.1 Copia il contenuto della cella di memoria riservata ad **a** nel registro A

   2.2 Copia il contenuto della cella di memoria riservata a **b** nel registro B

   2.3 Somma contenuto dei registri A e B

Esempio: più nel dettaglio

1. Leggi dal dispositivo di ingresso il valore delle variabili **a**
2. Somma il valore di **a** al valore di **b**
3. Salva il risultato parziale ottenuto

1. Scrivi nella cella di memoria centrale riservata al valore di **a** il valore letto dal registro dati della periferica.
   - Occorre la posizione di **a,b,c,d**
2. Somma il valore di **a** al valore di **b**
   2.1 Copia il contenuto della cella di memoria riservata ad **a** nel registro A
   2.2 Copia il contenuto della cella di memoria riservata a **b** nel registro B
   2.3 Somma contenuto dei registri A e B
3. Salva il risultato parziale, contenuto nel registro A, in una cella di memoria predisposta per il risultato (**z**).

Esempio: più nel dettaglio

1. Leggi dal dispositivo di ingresso il valore delle variabili **a**
2. Somma il valore di **a** al valore di **b**
3. Salva il risultato parziale ottenuto
4. Somma il valore di **c** al valore di **d**

4. Somma il valore di **c** al valore di **d**
   - 4.1 Copia il contenuto della cella di memoria riservata a **c** nel registro A
   - 4.2 Copia il contenuto della cella di memoria riservata a **d** nel registro B
   - 4.3 Somma il contenuto dei registri A e B

Esempio: più nel dettaglio

1. Leggi dal dispositivo di ingresso il valore delle variabili **a**
2. Somma il valore di **a** al valore di **b**
3. Salva il risultato parziale ottenuto
4. Somma il valore di **c** al valore di **d**
5. Moltiplica il risultato parziale appena ottenuto con quello precedentemente salvato

4. Somma il valore di **c** al valore di **d**
   - 4.1 Copia il contenuto della cella di memoria riservata a **c** nel registro A
   - 4.2 Copia il contenuto della cella di memoria riservata a **d** nel registro B
   - 4.3 Somma il contenuto dei registri A e B
5. Moltiplica
   - 5.1 Copia il contenuto della cella **z** nel registro B (**z** e B contengono ora **a + b**, mentre A contiene **c + d**)
   - 5.2 Moltiplica contenuto dei registri A e B

Esempio: più nel dettaglio

6. Scrivi sul dispositivo di uscita il risultato della valutazione complessiva

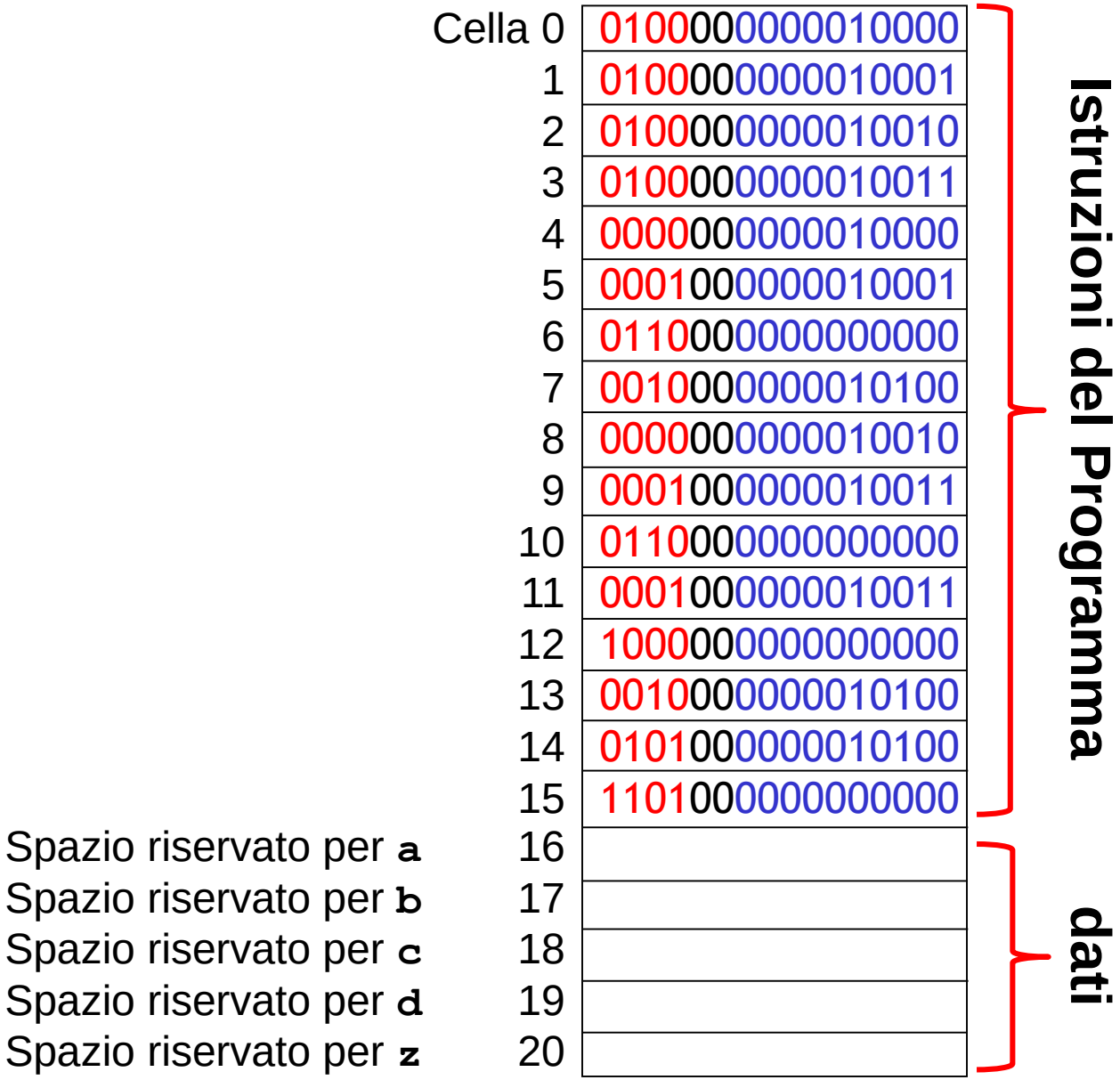
6. Scrivi sul dispositivo in uscita
   6.1 Memorizza il risultato calcolato (disponibile nel registro A) nella cella di memoria riservata a **z**
   6.2 Copia il contenuto della cella di memoria riservata a **z** nel registro dati della periferica di uscita

Esempio: più nel dettaglio

6. Scrivi sul dispositivo di uscita il risultato della valutazione complessiva
7. Termina l'esecuzione del programma

6. Scrivi sul dispositivo in uscita
   6.1 Memorizza il risultato calcolato (disponibile nel registro A) nella cella di memoria riservata a **z**
   6.2 Copia il contenuto della cella di memoria riservata a **z** nel registro dati della periferica di uscita
7. Manda il comando di Halt

Programma in Memoria Centrale



Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (`a`)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (`b`)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (`c`)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (`d`)
0000000000010000	**Carica** il contenuto della cella 16 (`a`) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (`b`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (`c`) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (`d`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (`z`) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (`z`) (ris. totale) output
1101000000000000	**Halt**

Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (`a`)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (`b`)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (`c`)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (`d`)
0000000000010000	**Carica** il contenuto della cella 16 (`a`) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (`b`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (`c`) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (`d`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (`z`) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (`z`) (ris. totale) output
1101000000000000	**Halt**

Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (`a`)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (`b`)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (`c`)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (`d`)
0000000000010000	**Carica** il contenuto della cella 16 (`a`) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (`b`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (`c`) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (`d`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (`z`) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (`z`) (ris. totale) output
1101000000000000	**Halt**

Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (**a**)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (**b**)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (**c**)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (**d**)
0000000000010000	**Carica** il contenuto della cella 16 (**a**) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (**b**) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (**z**) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (**c**) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (**d**) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (**z**) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (**z**) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (**z**) (ris. totale) output
1101000000000000	**Halt**

Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (**a**)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (**b**)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (**c**)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (**d**)
0000000000010000	**Carica** il contenuto della cella 16 (**a**) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (**b**) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (**z**) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (**c**) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (**d**) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (**z**) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (**z**) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (**z**) (ris. totale) output
1101000000000000	**Halt**

Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (`a`)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (`b`)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (`c`)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (`d`)
0000000000010000	**Carica** il contenuto della cella 16 (`a`) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (`b`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (`c`) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (`d`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (`z`) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (`z`) (ris. totale) output
1101000000000000	**Halt**

Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (`a`)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (`b`)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (`c`)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (`d`)
0000000000010000	**Carica** il contenuto della cella 16 (`a`) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (`b`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (`c`) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (`d`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (`z`) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (`z`) (ris. totale) output
1101000000000000	**Halt**

Forma Binaria del Programma

0100000000010000	**Leggi** un valore dall'input e mettilo nella cella 16 (`a`)
0100000000010001	**Leggi** un valore dall'input e mettilo nella cella 17 (`b`)
0100000000010010	**Leggi** un valore dall'input e mettilo nella cella 18 (`c`)
0100000000010011	**Leggi** un valore dall'input e mettilo nella cella 19 (`d`)
0000000000010000	**Carica** il contenuto della cella 16 (`a`) **nel registro A**
0001000000010001	**Carica** il contenuto della cella 17 (`b`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris.parziale)
0000000000010010	**Carica** il contenito della cella 18 (`c`) **nel registro A**
0001000000010011	**Carica** il contenito della cella 19 (`d`) **nel registro B**
0110000000000000	**Somma** i registri A e B
0001000000010100	**Carica** il contenuto della cella 20 (`z`) (ris. parziale) **in B**
1000000000000000	**Moltiplica** i registri A e B
0010000000010100	**Scarica** il contenuto di A nella cella 20 (`z`) (ris. totale)
0101000000010100	**Scrivi** il contenuto della cella 20 (`z`) (ris. totale) output
1101000000000000	**Halt**