Introduzione al Corso

Informatica, AA 2020/2021

Giacomo Boracchi

15 Settembre 2020

https://boracchi.faculty.polimi.it/

giacomo.boracchi@polimi.it

Ci presentiamo

Chi siamo

Giacomo Boracchi (giacomo.boracchi@polimi.it)

Matematico (Università Statale degli Studi di Milano 2004),

PhD in Information Technology (DEIB, Politecnico di Milano 2008)

Professore Associato dal 2019 al DEIB, Polimi (Computer Science)

Nella mia ricerca mi occupo di modelli e metodi matematici e statistici per:

Analisi/elaborazione di immagini

Machine Learning (unsupervised learning), change and anomaly detection

Chi siamo

Giacomo Boracchi (docente)

- Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano
- homepage: https://boracchi.faculty.polimi.it/
- email giacomo.boracchi@polimi.it
- Webex Room: https://politecnicomilano.webex.com/join/giacomo.boracchi
- ricevimento studenti:
  - **Su appuntamento**, da prendere via email.
- Ufficio 157 via Ponzio 34/5, Milano.
- tel 02 2399 3467

Chi siamo

Giacomo Boracchi (docente)

- Dipartimento di Elettronica e Informazione e Bioingegneria, (DEIB) Politecnico di Milano
- homepage: http://home.deib.polimi.it/boracchi/
- email giacomo.boracchi@polimi.it
- ricevimento studenti:

  Su appuntamento, da **prendere via email.**
- Ufficio 157 via Ponzio 34/5, Milano.
- tel 02 2399 3467

Chi siamo

Stefano Marelli (esercitatore e responsabile di laboratorio)

- email: stefano1.marelli@polimi.it

I materiali di esercitazioni e laboratorio saranno caricati sulla pagina del corso

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

Il Corso

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

Organizzazione

Lezioni: 28 ore

Esercitazioni: 24 ore

Laboratori: 16 ore

Lezioni in presenza: il Martedì dalle 9.00 alle 11:00 in aula B.1.5.

Lezioni online: il Venerdì dalle 14.30 alle 17:00

Controllate sempre il **calendario** del corso:

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

Dove troverete anche indicazioni su chi terrà ogni lezione ed esercitazione.

La Pagina del Corso

La pagina del corso è

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

Troverete:

- Materiale didattico usato a lezione (queste slides sono da considerare **un supporto** allo studio)
- Alcuni temi d'esame
- Calendario del corso (lezioni, esercitazioni, laboratori)
- Avvisi, esiti esami/prove intermedie

Controllate sempre il calendario sul sito per vedere a quale aula webex collegarvi:

https://boracchi.faculty.polimi.it/teaching/InfoICMRCalendar.htm

I link alle registrazioni verranno riportati nel calendario

È possibile che nei prossimi giorni ci sia qualche cambiamento al calendario

I Laboratori

Nei laboratori vi sarà richiesto di **sviluppare autonomamente** gli elaborati.

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.

A laboratorio potrete usare i PC dell'aula informatizzata

Vi saranno comunque date informazioni per installare Matlab sul vostro PC

Lezioni ed Esercitazioni

How to..

Modalità di Insegnamento

Due consigli:

sfruttate al massimo la presenza (anche virtuale) di docenti ed esercitatori.

Riguardate regolarmente almeno gli esercizi svolti

Molto bene l'interazione docente-studente:

Farò domande: rispondete (sia da casa che dall'aula)!

Chiedete le cose che non vi sono chiare!

Durante le esercitazioni in modalità blended si userà **prevalentemente la lavagna (non tutte le slides verranno mostrate)**

Modalità di Insegnamento

Due consigli:

sfruttate al massimo la presenza (anche virtuale) di docenti ed esercitatori.

Riguardate regolarmente almeno gli esercizi svolti

Molto bene l'inter

Farò domande: ris

Chiedete le cose

Non vale arrendersi: non si esce prima della fine dell'esercitazione (o della pausa)!

Durante le esercitazioni in modalità blended si userà **prevalentemente la lavagna (non tutte le slides verranno mostrate)**

Esercizi a lezione:

Farò **molti esercizi** durante la lezione

Spesso svolti al PC o «pasticciando» le slides

Non sempre ci saranno le soluzioni nei materiali online

Consiglio di **non programmare mentre seguite le lezioni**

Prendete appunti! Non fate affidamento sulle registrazioni (non potete certo ri-guardarle tutte) o sulle slides (potrebbero contenere solo parte delle cose che spiego)

Partecipazione alle Lezioni

Mi auspico la massima interazione durante lezioni/esercitazioni online/blended!

Potete comunicare

Togliendo il muto e ponendo educatamente una domanda (mi aspetto sappiate scegliere il momento opportuno)

Scrivendo nella chat a tutta la classe (o al docente)

Cisco Webex Meetings | Meeting Info | Hide menu bar

Connected

File Edit Share View Audio & Video Participant Meeting Help

Chat

GB

Assicuratevi di avere il microfono spento durate la lezione

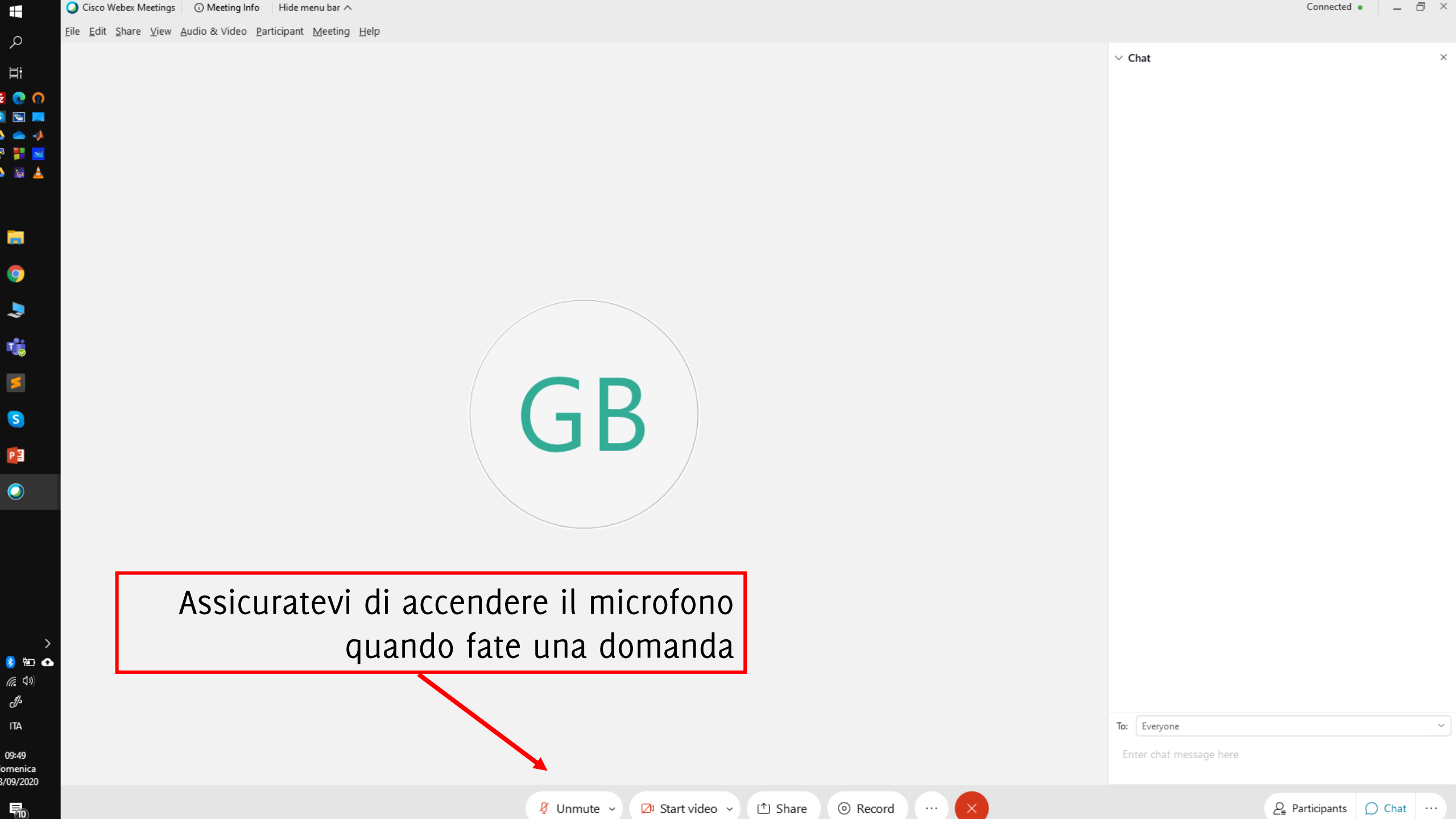
To: Everyone

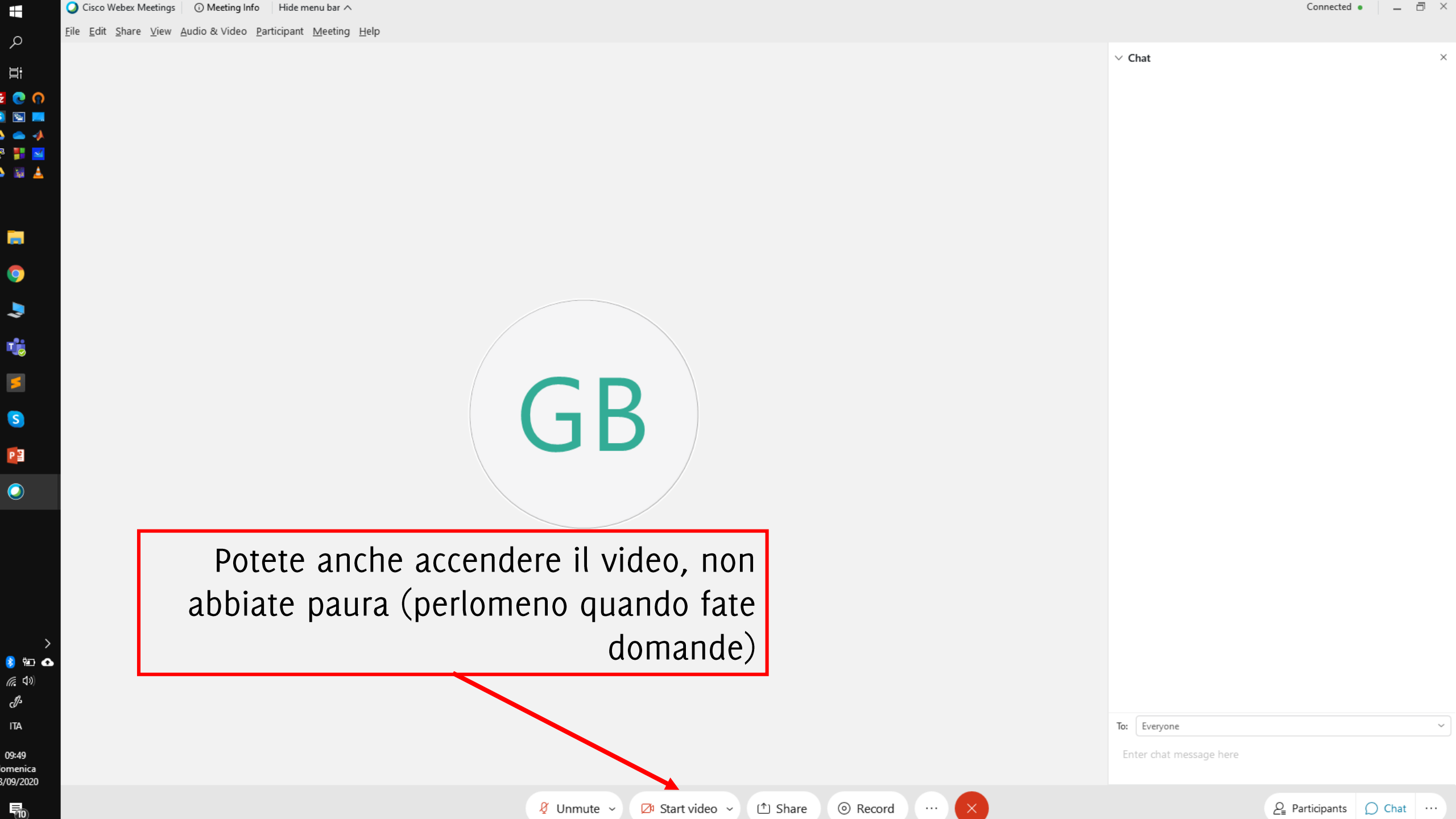
Enter chat message here

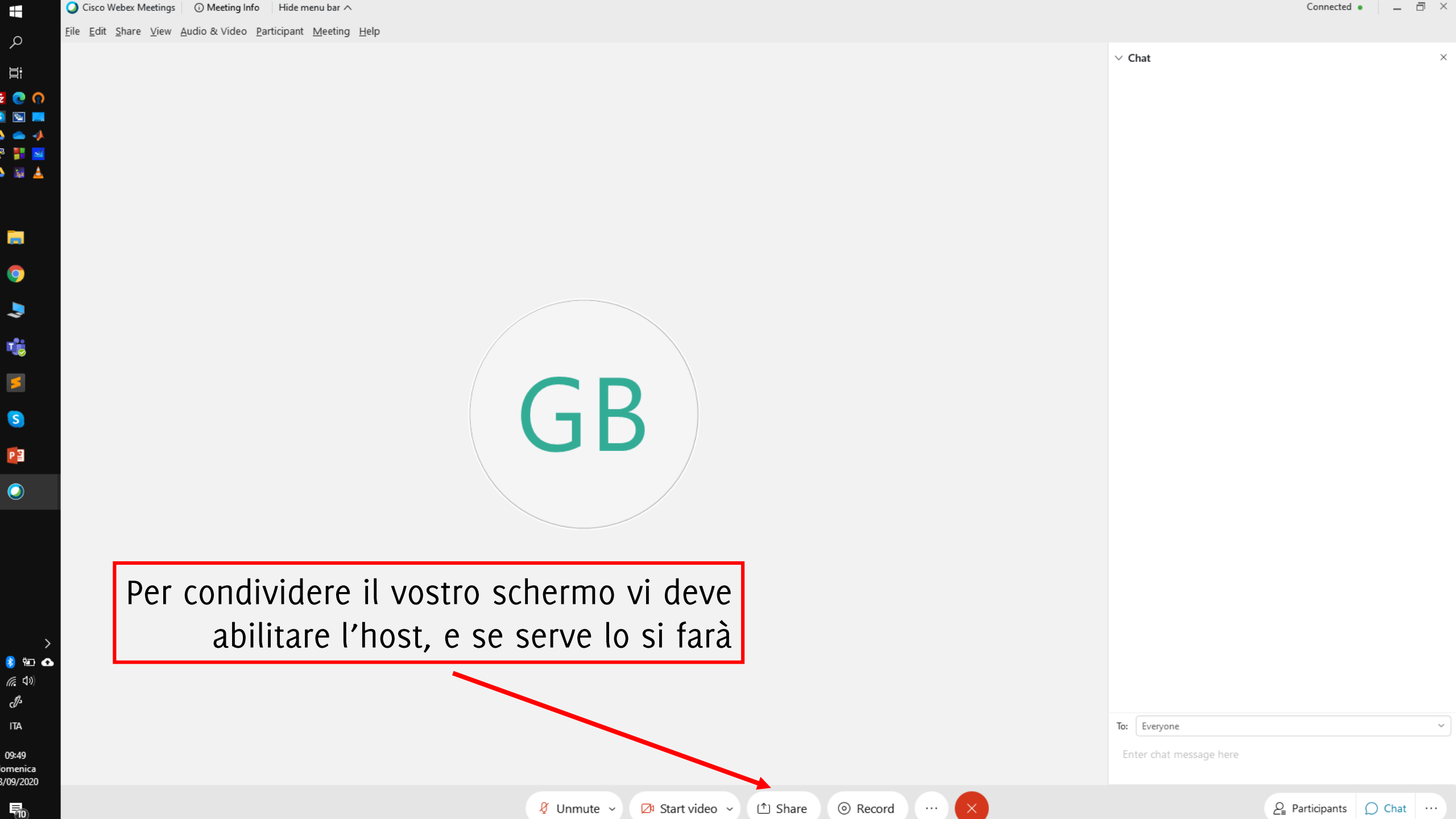
Unmute | Start video | Share | Record | ... | Participants | Chat | ...

ITA

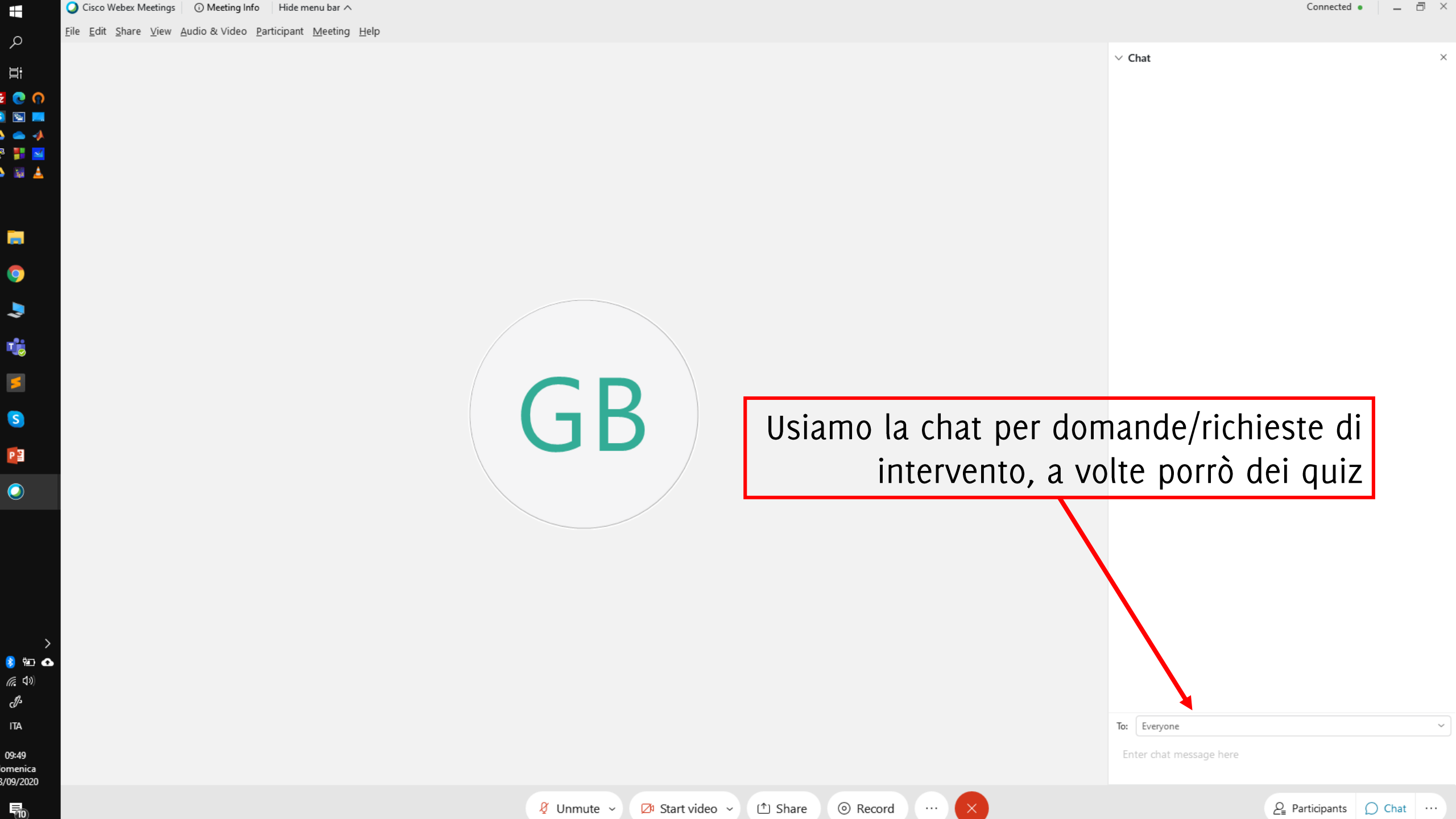
09:49
omenica
3/09/2020


Cisco Webex Meetings
Meeting Info
Hide menu bar
Connected
File Edit Share View Audio & Video Participant Meeting Help
Chat
GB
Assicuratevi di accendere il microfono quando fate una domanda
To: Everyone
Enter chat message here
Unmute
Start video
Share
Record
Participants
Chat
ITA
09:49
3/09/2020


Cisco Webex Meetings
Meeting Info
Hide menu bar
Connected
File Edit Share View Audio & Video Participant Meeting Help
Chat
GB
Potete anche accendere il video, non abbiate paura (perlomeno quando fate domande)
To: Everyone
Enter chat message here
Unmute
Start video
Share
Record
Participants
Chat
ITA
09:49
omenica
3/09/2020


Cisco Webex Meetings
Meeting Info
Hide menu bar
Connected
File Edit Share View Audio & Video Participant Meeting Help
Chat
GB
Per condividere il vostro schermo vi deve abilitare l'host, e se serve lo si farà
To: Everyone
Enter chat message here
Unmute
Start video
Share
Record
Participants
Chat
ITA
09:49
3/09/2020


Cisco Webex Meetings
Meeting Info
Hide menu bar
Connected
File Edit Share View Audio & Video Participant Meeting Help
Chat
GB
Devo registrare le lezioni, ricordatemelo!
To: Everyone
Enter chat message here
Unmute
Start video
Share
Record
Participants
Chat
ITA
09:49
3/09/2020


Cisco Webex Meetings
Meeting Info
Hide menu bar
Connected
File Edit Share View Audio & Video Participant Meeting Help
Chat
GB
Usiamo la chat per domande/richieste di intervento, a volte porrò dei quiz
To: Everyone
Enter chat message here
Unmute
Start video
Share
Record
Participants
Chat
ITA
09:49
3/09/2020

Richieste di Chiarimento

Richieste di Chiarimento

È bene che le lezioni siano quanto più interattive possibile.

Domande e richieste di chiarimenti sono sempre ben accette:

- Domande via chat
- Domandando direttamente a voce/video

Potete anche rivolgere domande via mail.

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

Secure | https://mail.google.com/mail/u/0/#all/159e546f3c53b2df

Google

Gmail

COMPOSE

Inbox (12)
Starred
Important
Sent Mail
Drafts (5)
All Mail
Spam (3)
Bin
Categories
[Gipi] (361)
[Lista PD] (450)
[matlab notifications...
Gilardoni
polimi.it (4,547)
smafsoft.it (1,184)
master-information
More

Move to Inbox | More

Domanda tde

Inviata da Polimi x | polimi.it x

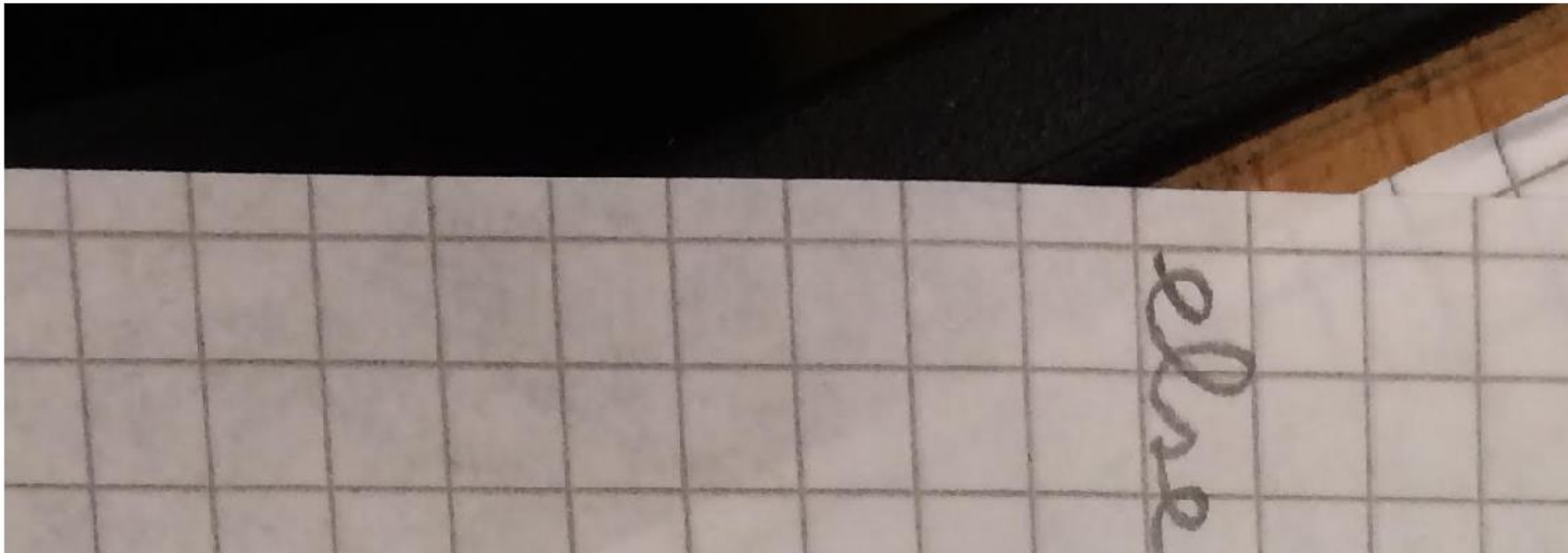
[redacted]
to Francesco, Giacomo

Buona sera,

È corretta questa soluzione per il punto 2 dell'esercizio 2 del tde 28/01/13?

Cordiali saluti

[redacted]



Esercizio 3 (4 punti)

Dati i seguenti due numeri in codifica IEEE 754 (virgola mobile, il bit più a sinistra è ovviamente il bit di segno)

```
A = S:0 E:01111111 M:00100000000000000000000
B = S:1 E:01111100 M:00100000000000000000000
```

Calcolare a quanto equivale la divisione A/B (in decimale).

Soluzione

I numeri A e B si differenziano solo per esponente e segno. Si può quindi calcolare la divisione A/B prendendo in considerazione solo gli esponenti:

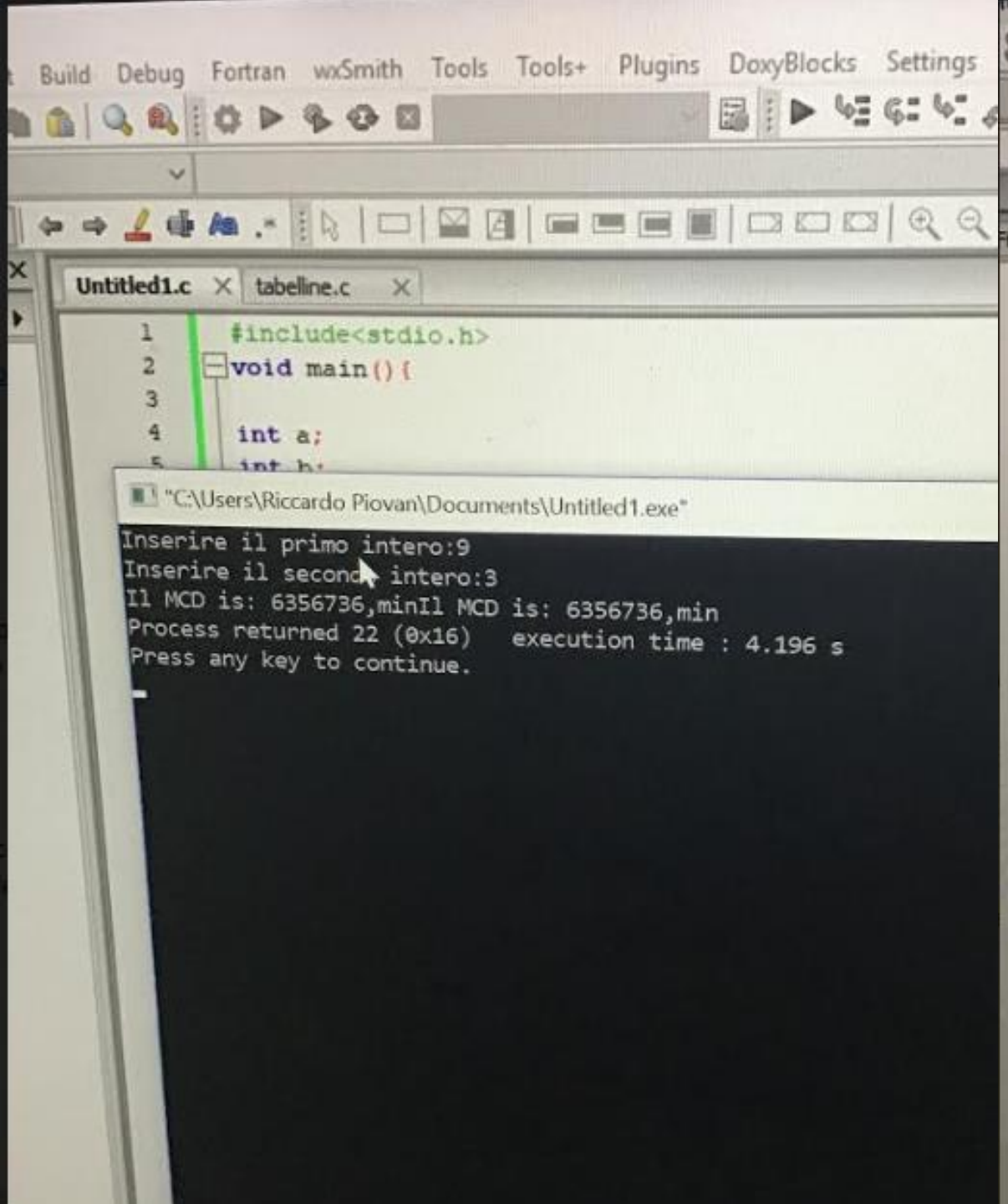
```
A = - B * 2^3
```

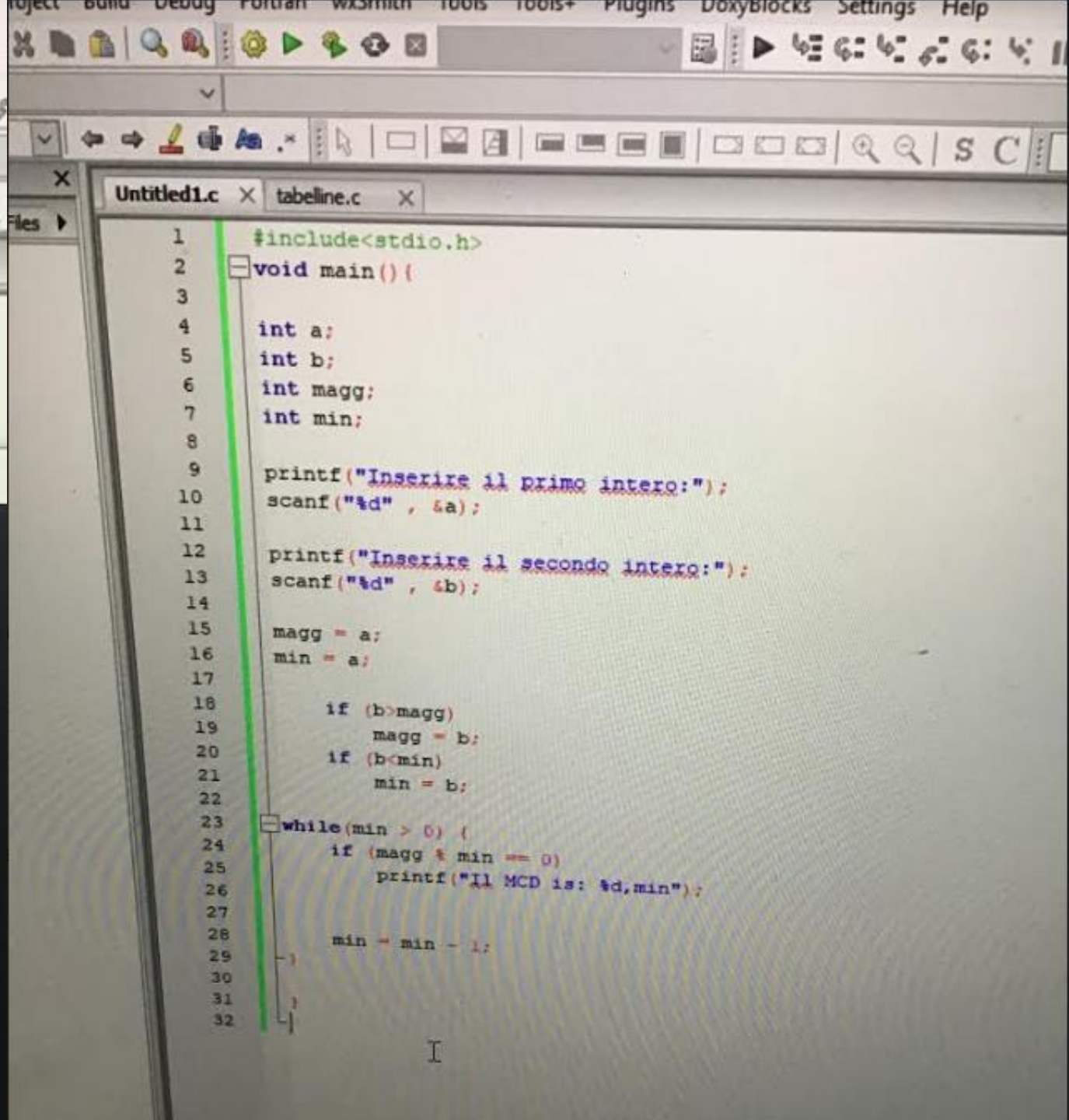
Quindi:

```
A/B = - 8
```

Si può anche notare che:

```
A =       (1 + 2^(-3))           =   1.125
B = -1 * (1 + 2^(-3)) * 2^(-3)   = - 0.140625
```


Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings
Untitled1.c
tabelline.c
1 #include<stdio.h>
2 void main(){
3
4 int a;
"C:\Users\Riccardo Piovan\Documents\Untitled1.exe"
Inserire il primo intero:9
Inserire il secondo intero:3
Il MCD is: 6356736,minIl MCD is: 6356736,min
Process returned 22 (0x16) execution time : 4.196 s
Press any key to continue.


Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Untitled1.c
tabelline.c
1 #include<stdio.h>
2 void main(){
3
4 int a;
5 int b;
6 int magg;
7 int min;
8
9 printf("Inserire il primo intero:");
10 scanf("%d" , &a);
11
12 printf("Inserire il secondo intero:");
13 scanf("%d" , &b);
14
15 magg = a;
16 min = a;
17
18 if (b>magg)
19 magg = b;
20 if (b<min)
21 min = b;
22
23 while(min > 0) {
24 if (magg % min == 0)
25 printf("Il MCD is: %d,min");
26
27
28 min = min - 1;
29 }
30
31 }
32

Richieste via mail

Quando scrivete una mail per una richiesta di chiarimento

-Dite di che corso siete (tengo un altro corso di informatica!)

-Allegate Il codice sorgente, ripulito e commentato

-Allegate il testo dell'esercizio (no foto)

-Un breve commento che spiega cosa non funziona e i tentativi che avete fatto.

L'Esame

Modalità di Verifica

Solo appelli regolari (niente prove intermedie):

- **Ci sono 5 appelli regolari, con date disponibili nel calendario del facoltà / sistema online. N**essun appello oltre a questi.
- Il laboratorio non sarà valutato
- Vi verrà richiesto **di risolvere dei programmi in Matlab al computer**

Gli studenti potranno ricevere **punti bonus** per:

- La loro partecipazione attiva a lezioni / esercitazioni / laboratori
- Partecipazione a contest presentati nel corso delle lezioni

L'esame orale non è previsto se non a discrezione del docente

Studenti con OFA in matematica (o totale) non possono sostenere le prove prima di aver passato l'OFA

Esame online

Press F11 to exit full screen

Informatica A 28/8/2020

La fase 1 si compone di DUE ESERCIZI C E DUE QUERY SQL.

- Ogni campo risposta puo' contenere AL MASSIMO 140 RIGHE.
Controllate che il vostro codice sia in questo limite e, nel caso non lo fosse, compattate il codice. In casi estremi potete non includere le funzioni ausiliarie che vi abbiamo fornito (esercizio delle liste) nelle soluzioni che caricate.

- potete sottomettere UNA SOLA RISPOSTA.

- e' necessario inserire almeno un carattere nel campo "risposta" di ogni domanda per poter sottomettere il form.

- TEMPO A DISPOSIZIONE 1H20

...

Hi Giacomo, when you submit this form, the owner will be able to see your name and email address.

* Required

1. Selezionare la TERZULTIMA (cioè la SESTA) cifra del vostro codice persona 10XXXXXX *

- ○ 0 oppure 1 oppure 3
- ○ 2 oppure 4
- ◉ 5 oppure 6 oppure 9
- ○ 7 oppure 8

2. Si sviluppi una funzione colonneGrandi che prende in ingresso una matrice quadrata di interi B che ha r righe e c colonne (r e c possono essere inferiori alla dimensione con cui B viene dichiarata), e restituisce al programma chiamante un vettore v di c - 1 elementi che contiene, nella posizione i-sima il numero di elementi della colonna i-sima che sono maggiori della media

Contenuti del Corso

Argomenti Trattati

- Introduzione all'informatica
- Algoritmi
- Aritmetica Binaria
- Programmazione in linguaggio Matlab
- Argomenti avanzati di programmazione

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.

Cosa imparerete:

Gli **elementi fondamentali** ed i **principi** che regolano il funzionamento di un **sistema informatico**

Alcune nozioni di base sulla **codifica binaria** e l'algebra di Boole

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 **Matlab**

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 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»]

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:

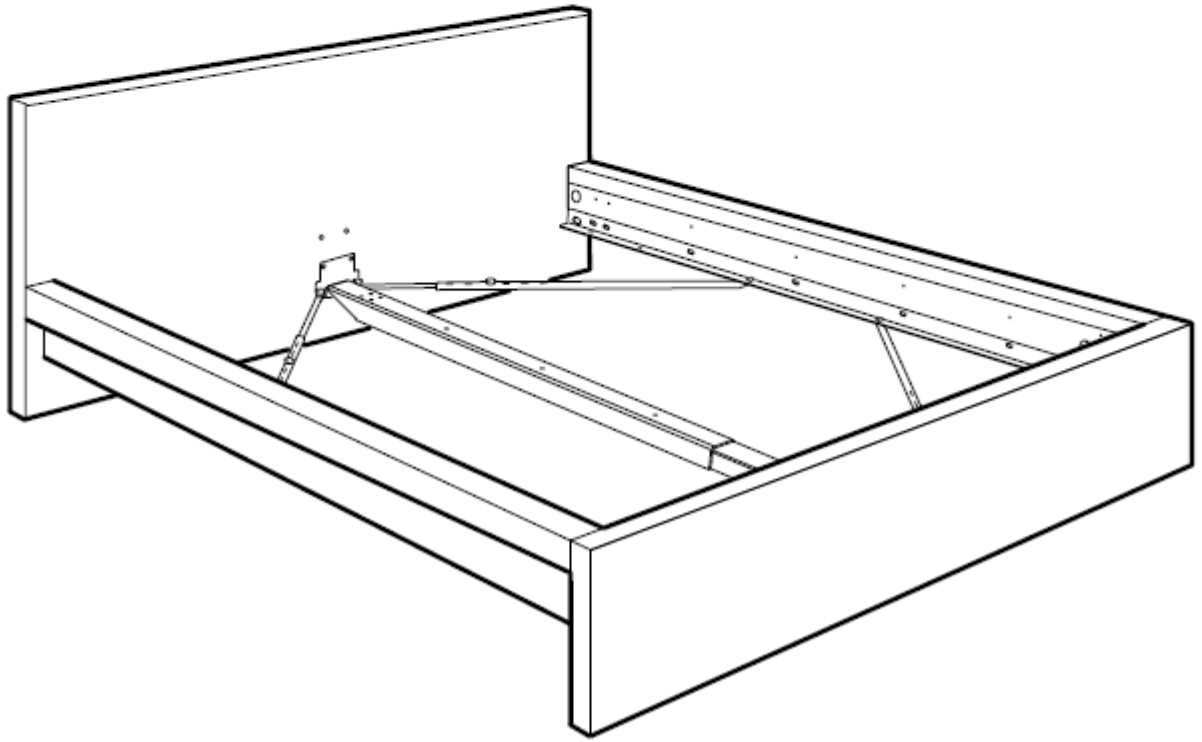
essere **comprensibili** senza ambiguità

essere **eseguibili** da uno strumento automatico: l'esecutore

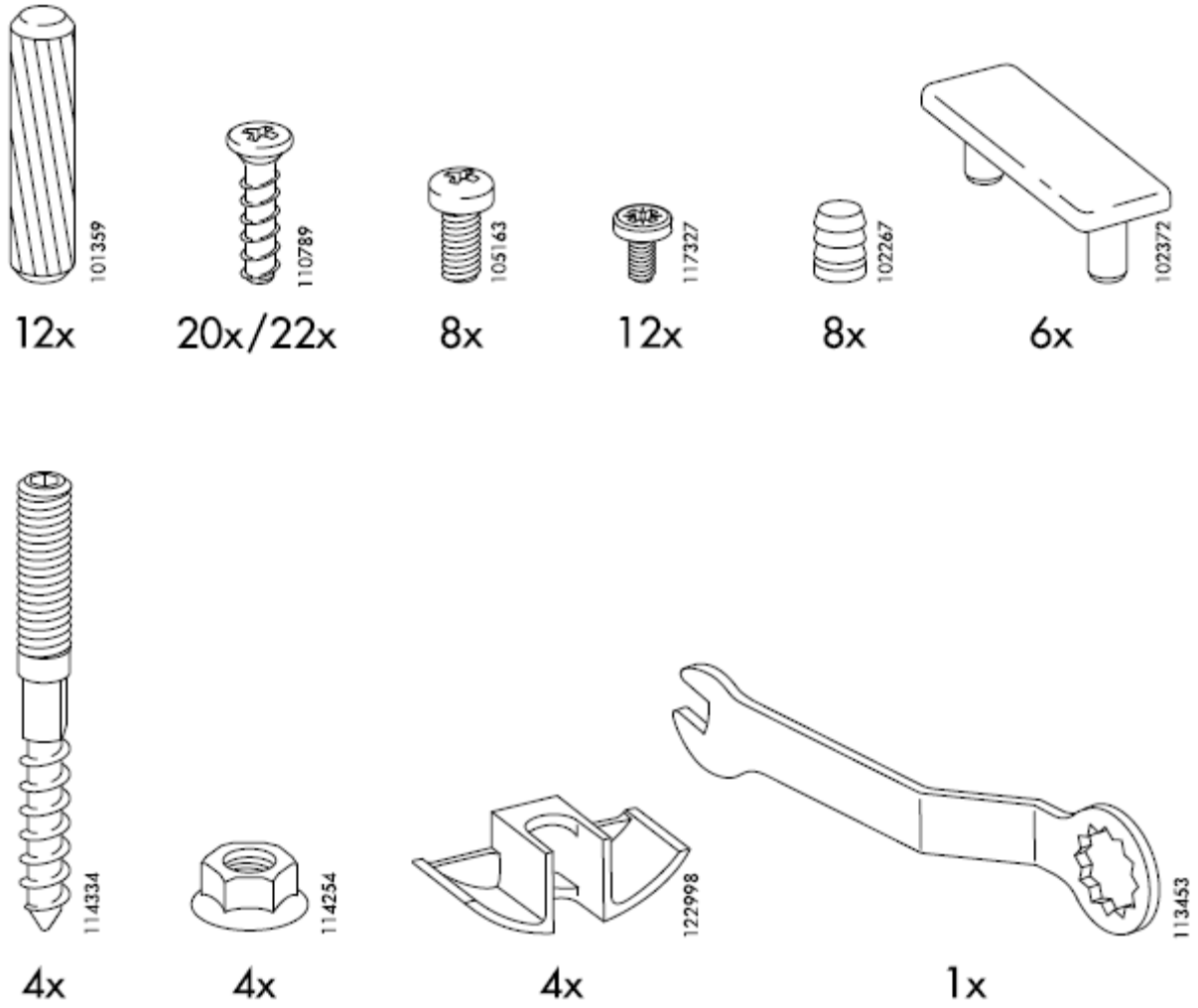
portare a realizzare un compito in **tempo finito** (devono contenere un numero finito di passi, ciascuno eseguibile in tempo finito)

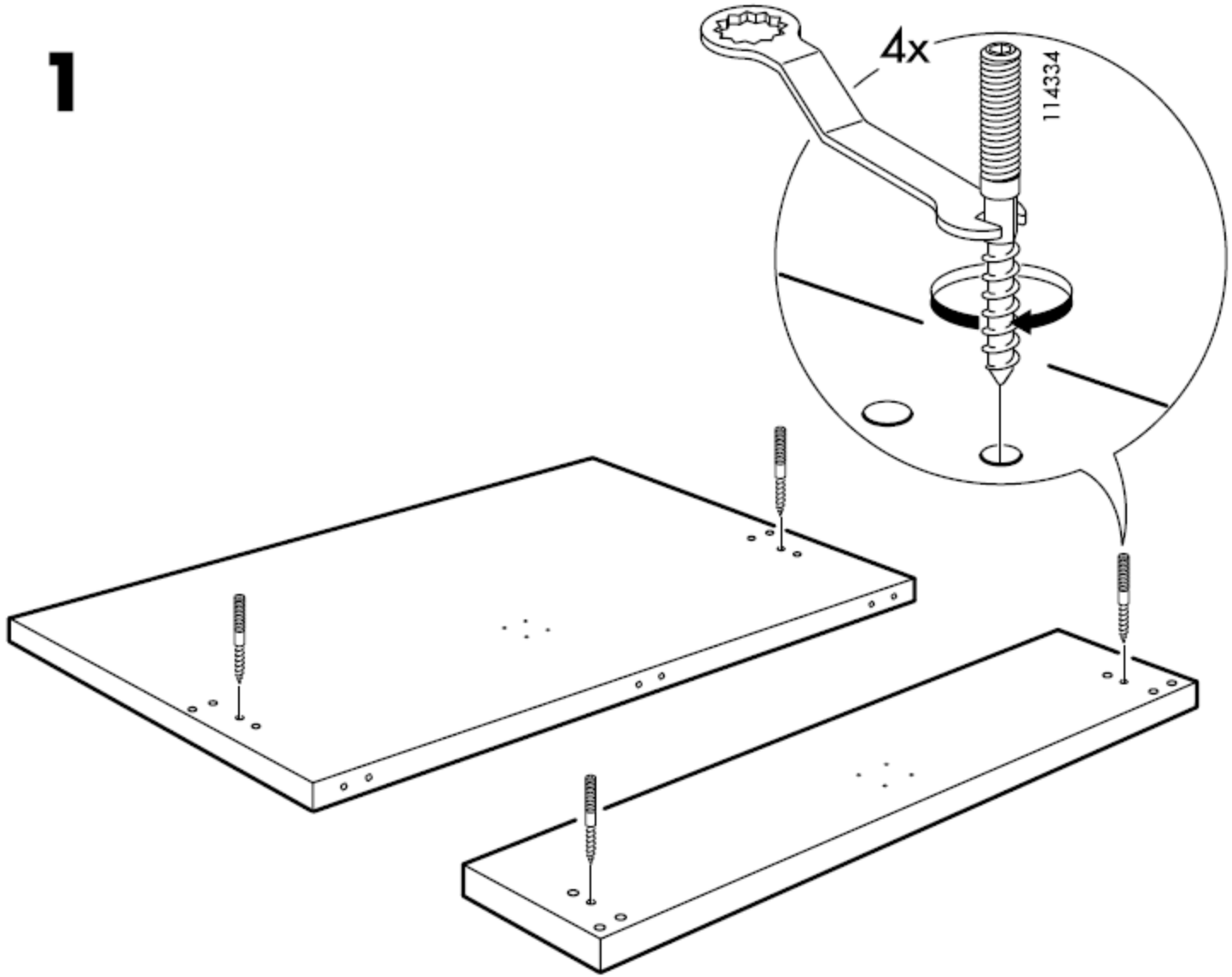
…ora vedermo un esempio di algoritmo in cui vi sarà capitato di essere esecutori

MALM

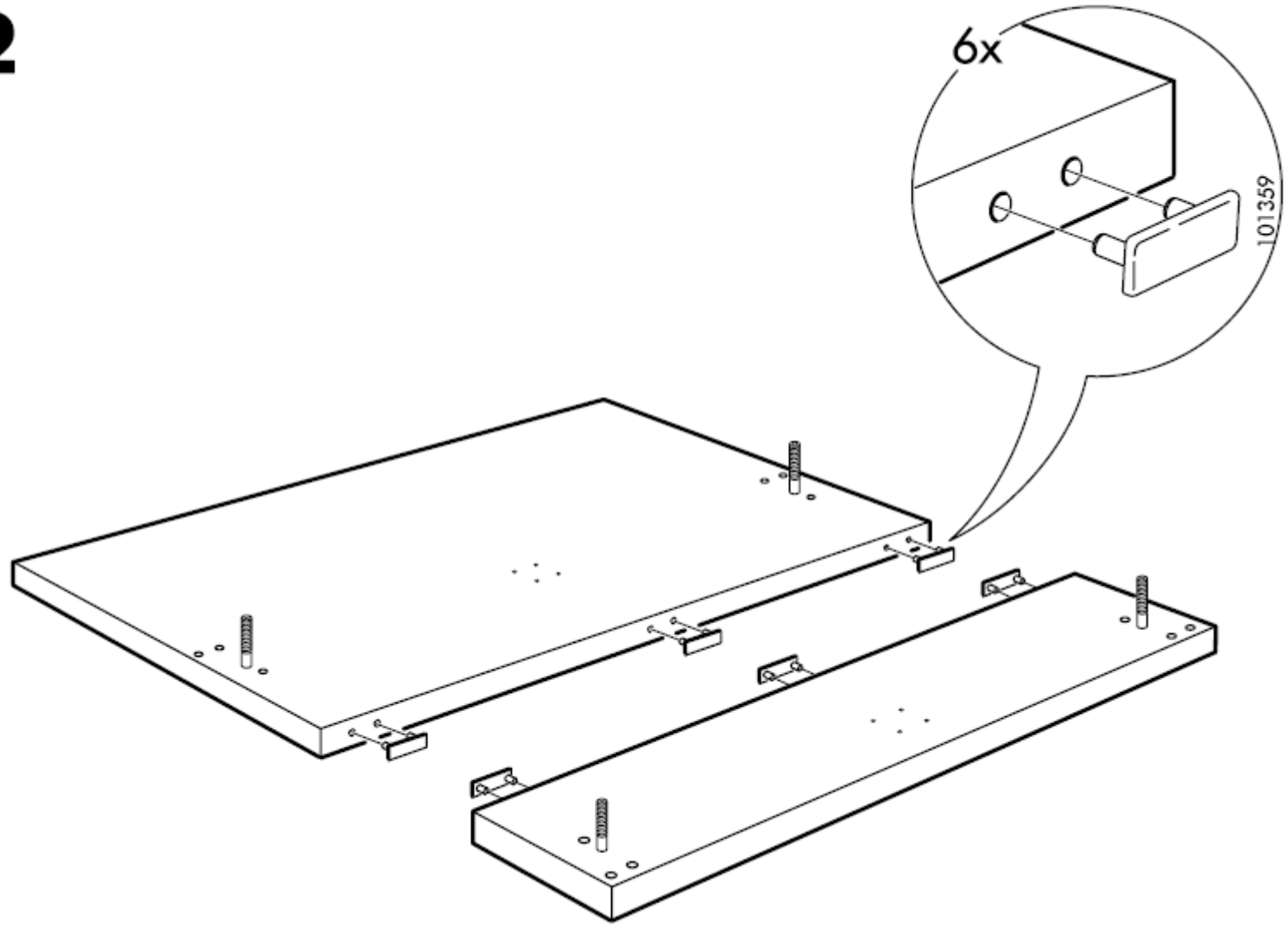


IKEA®
Design and Quality
IKEA of Sweden


101359
12x
110789
20x/22x
105163
8x
117327
12x
102267
8x
102372
6x
114334
4x
114254
4x
122998
4x
113453
1x


1
4x
114334

2



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)

Il Linguaggio

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

Linguaggi di Programmazione

Programmazione a Basso / Alto Livello

Linguaggio macchina: poche istruzioni, difficile codificare algoritmi e interpretare il codice

- Linguaggio preciso.
- Completo controllo delle risorse.

Esempio di Linguaggio a basso livello

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**

Programmazione a Basso / Alto Livello

Linguaggio macchina: poche istruzioni, difficile codificare algoritmi e interpretare il codice

- Linguaggio preciso.
- Completo controllo delle risorse.

Linguaggi di alto livello: linguaggi più comprensibili per l'uomo.

- Linguaggio preciso e sintetico
- Riferimenti simbolici
- Esprimere istruzioni in linguaggio vicino a quello naturale.

Esempio di linguaggio ad alto livello

```
value = 1000;
year = 0;
while value < 2000
    value = value * 1.08
    year = year + 1;
    fprintf('%g years: $%g\n', year,value)
end
```

(calcola gli interessi fino al raddoppio del capital con un tasso dell’ 8%)

Programmazione a Basso / Alto Livello

Linguaggio macchina: poche istruzioni, difficile codificare algoritmi e interpretare il codice

- Linguaggio preciso.
- Completo controllo delle risorse.

Linguaggi di alto livello: linguaggi più comprensibili per l'uomo.

- Linguaggio preciso e sintetico
- Riferimenti simbolici
- Esprimere istruzioni in linguaggio vicino a quello naturale.

La traduzione dal linguaggio ad alto livello al linguaggio macchina è eseguita da un altro programma, il **compilatore** o dall'**interprete**

La Sequenzialità

Esempio: algoritmo per andare in Università

Mi alzo quando suona la sveglia

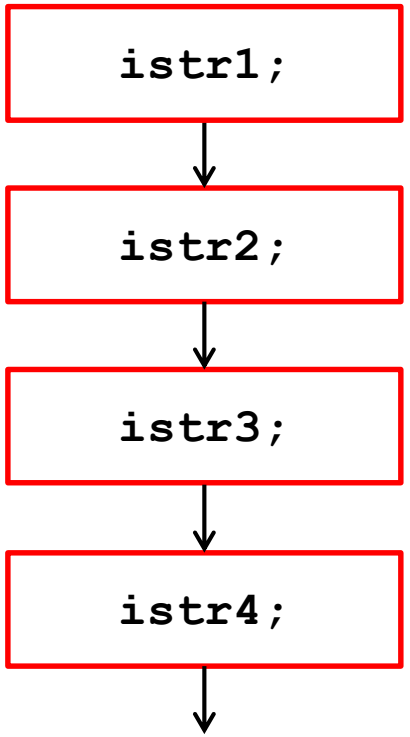
Mi dirigo verso la cucina

Mangio e bevo un caffè

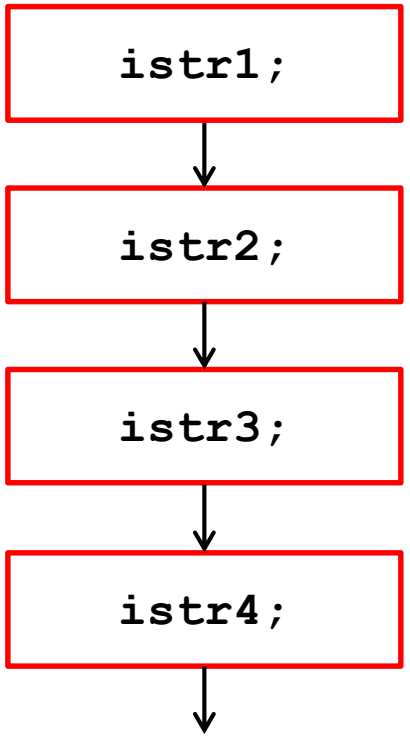
Mi lavo

Mi vesto

La sequenzialità



La sequenzialità

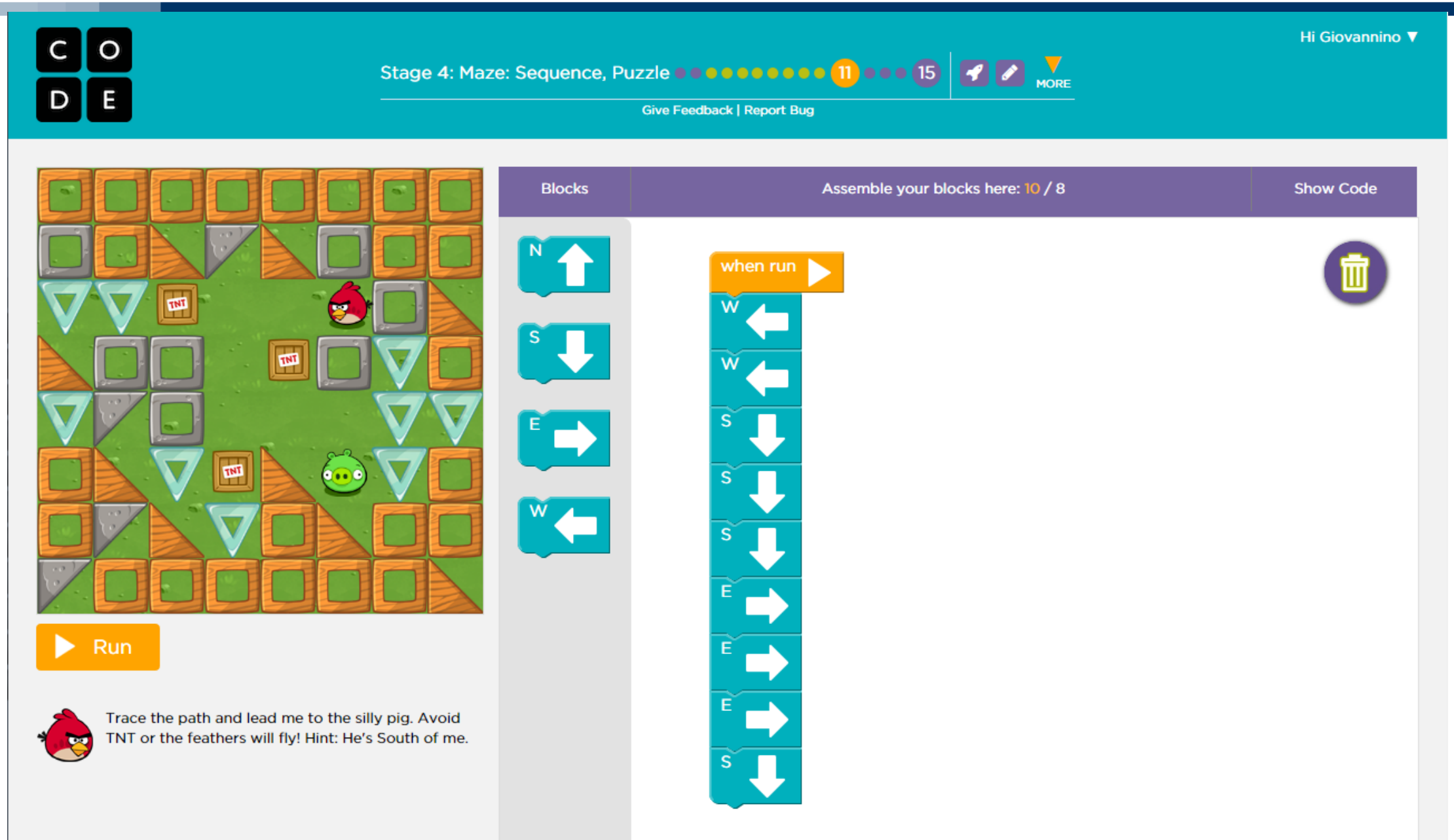



```
istr1;
istr2;
istr3;
istr4;
...
```

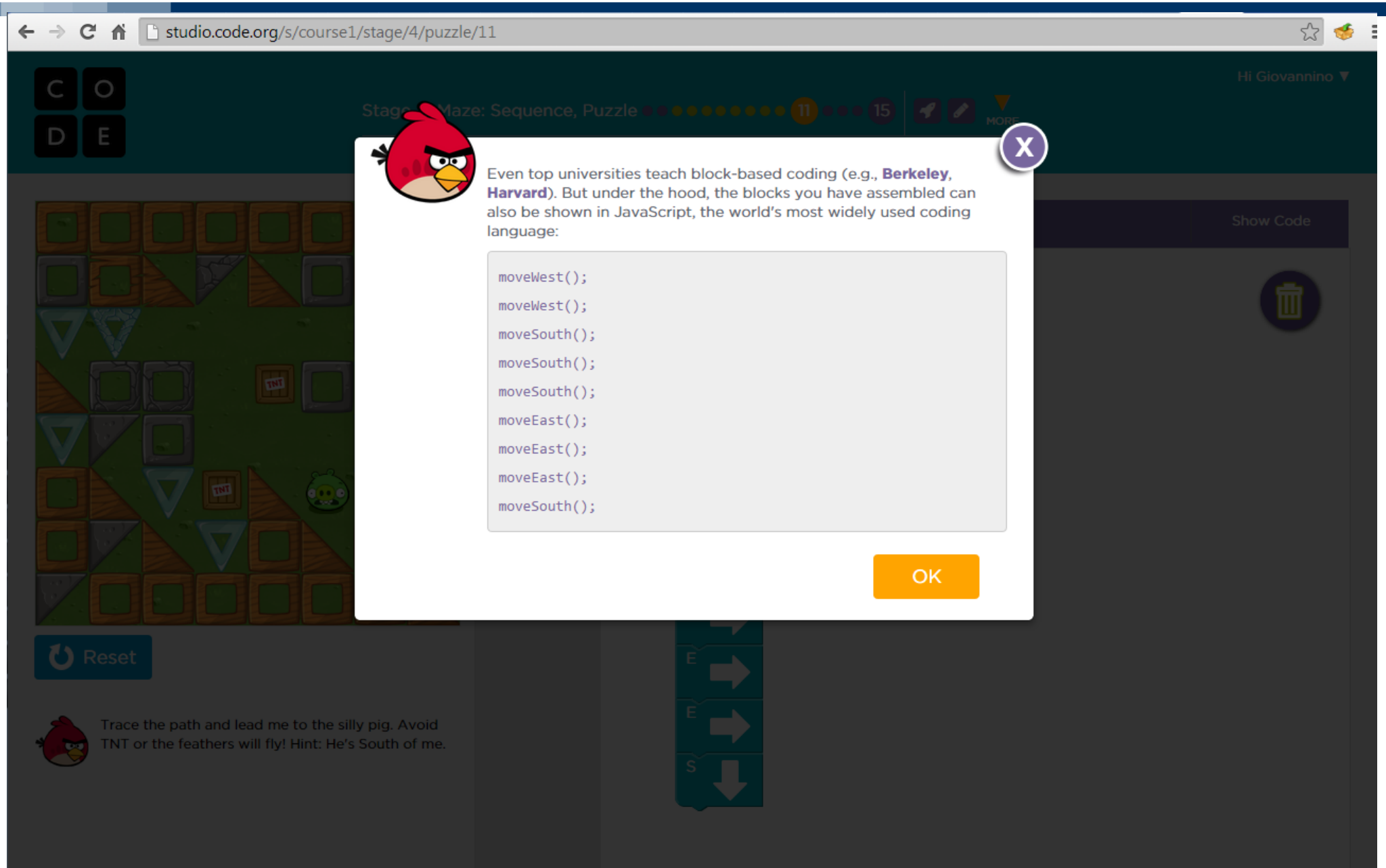
Le istruzioni vengono eseguite dalla prima all'ultima.

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

La sequenzialità per i bimbi su code.org



La sequenzialità per i bimbi su code.org



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

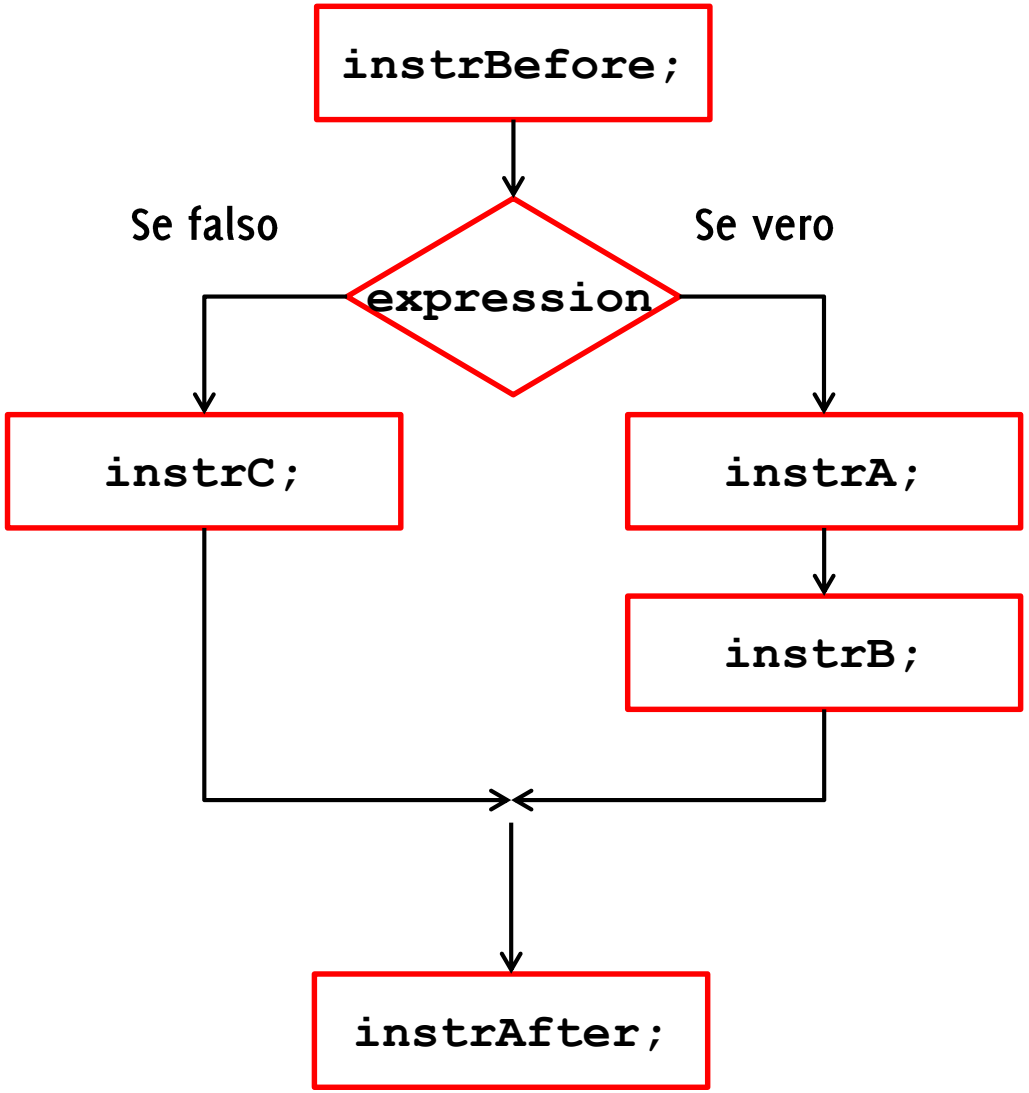
Se devo mangiare in fuori

- prendo il pranzo

Altrimenti

- prendo uno snack per metà mattina

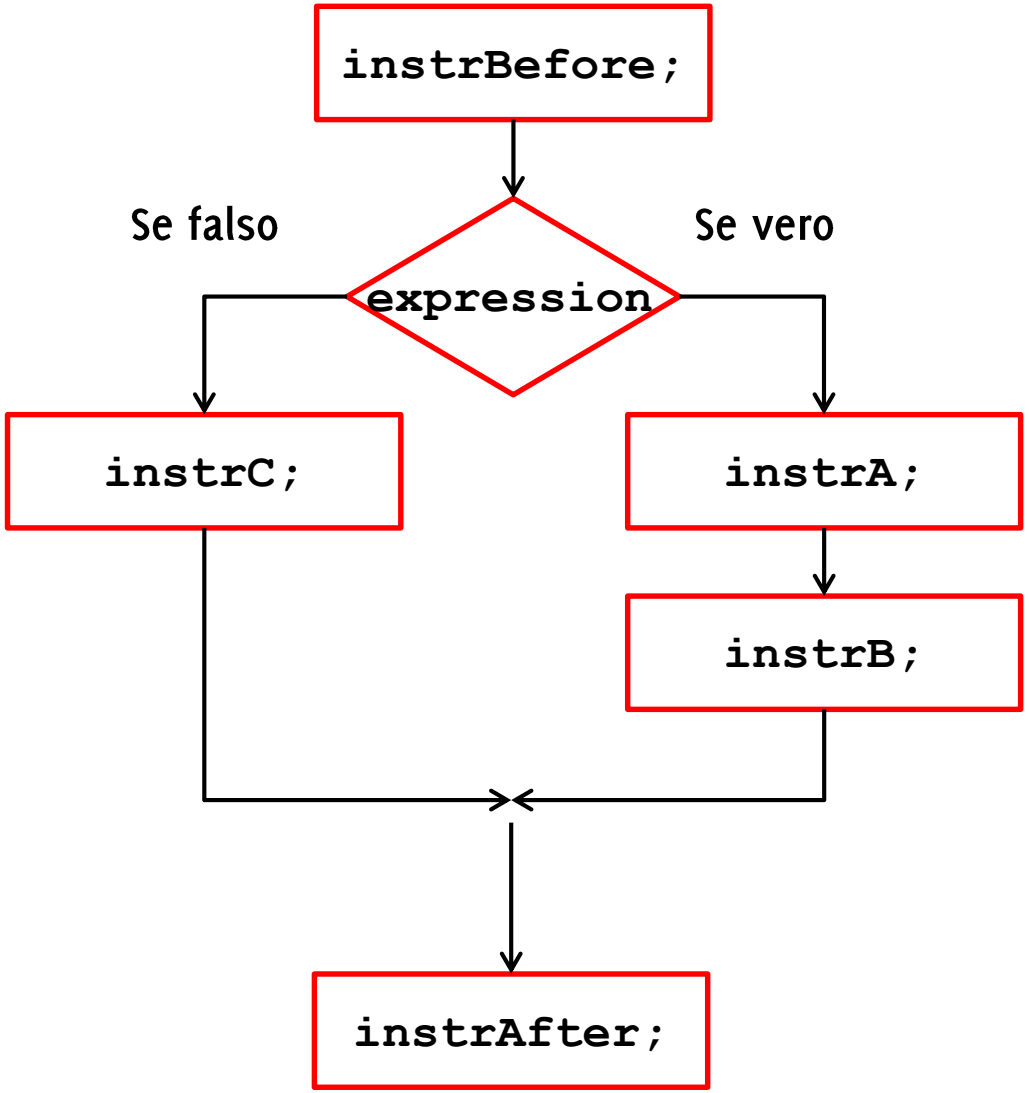
Costrutto Condizionale:



ore si

io il ramo
A; e

Costrutto Condizionale:

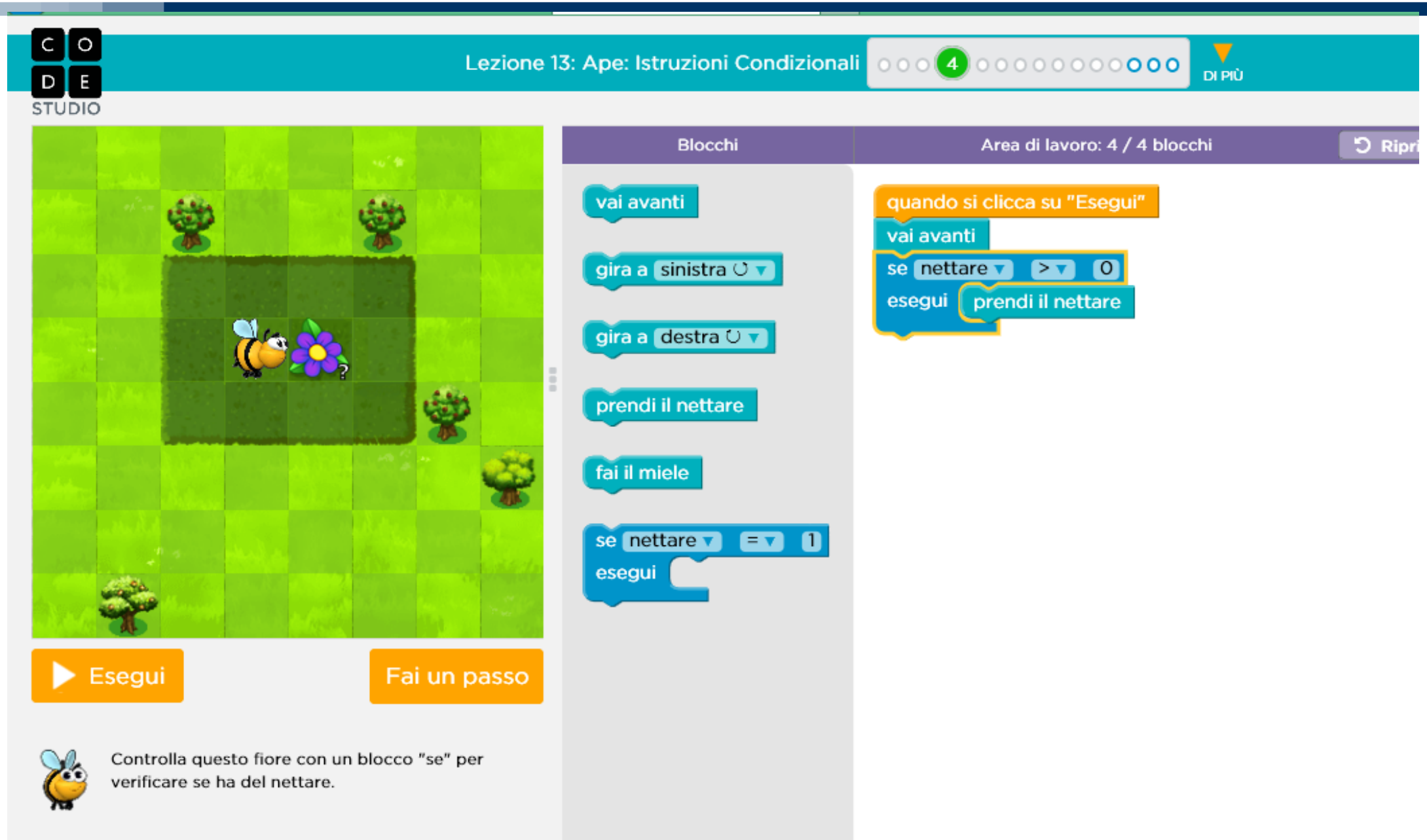


Dopo aver eseguito **`instrBefore`** si valuta **`expression`**

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

Altrimenti eseguo **`instrC;`**

Il costrutto condizionale su code.org



Il costrutto condizionale su code.org

Esempio: algoritmo per andare in Università

Mi alzo quando suona la sveglia

Mi dirigo verso la cucina

Mangio e bevo un caffè

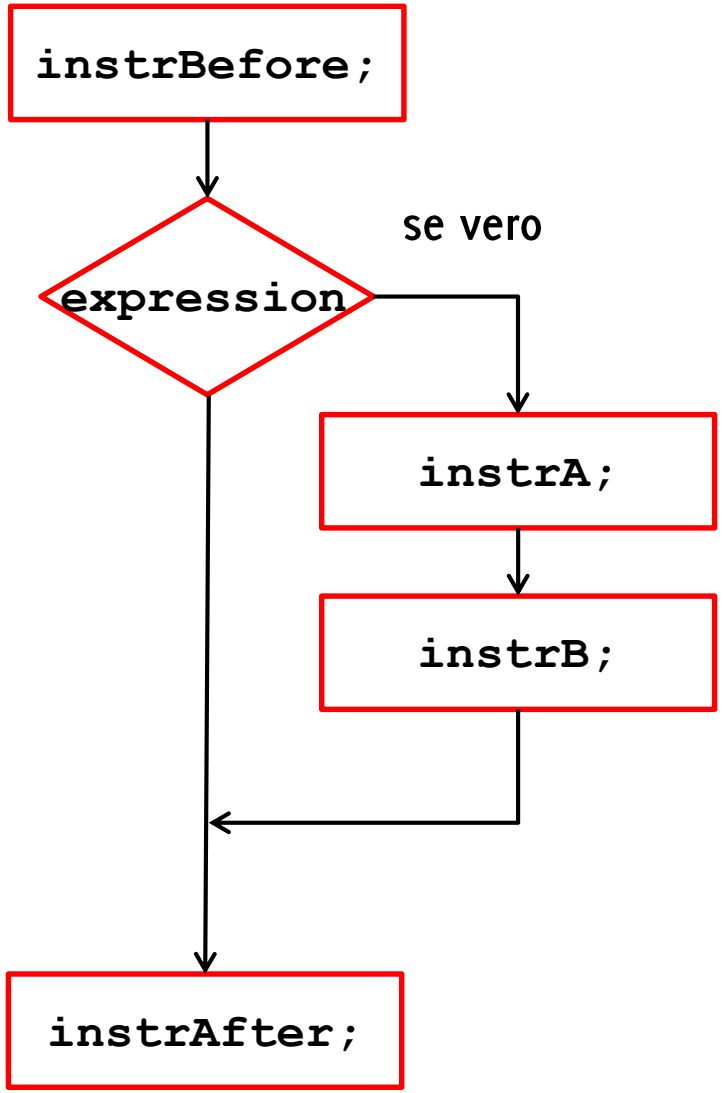
Mi lavo e mi vesto

Se c'è il laboratorio di informatica

- Prendo il laptop

Esco di casa

Costrutto Condizionale:



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

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

Ripeti: piove ancora?

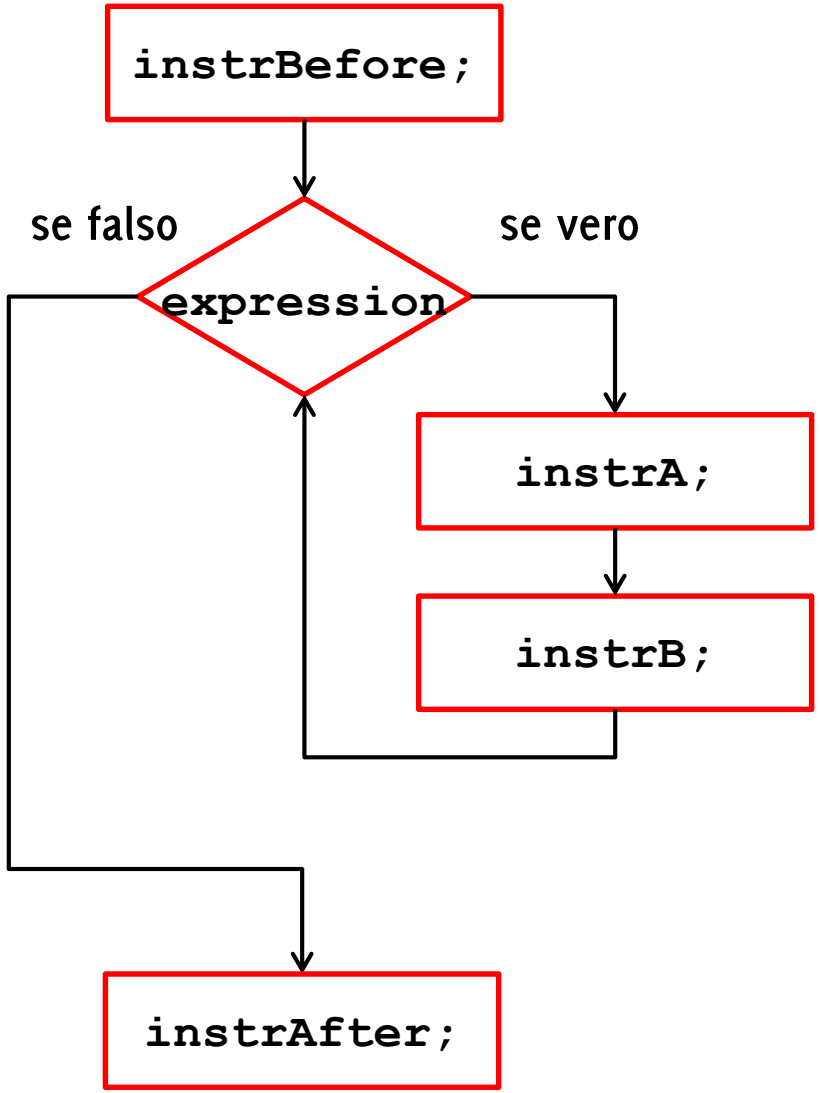
- Aspetta a casa

Vado in stazione

Sali sul treno

Arrivi a lezione, wow

Costrutto Iterativo

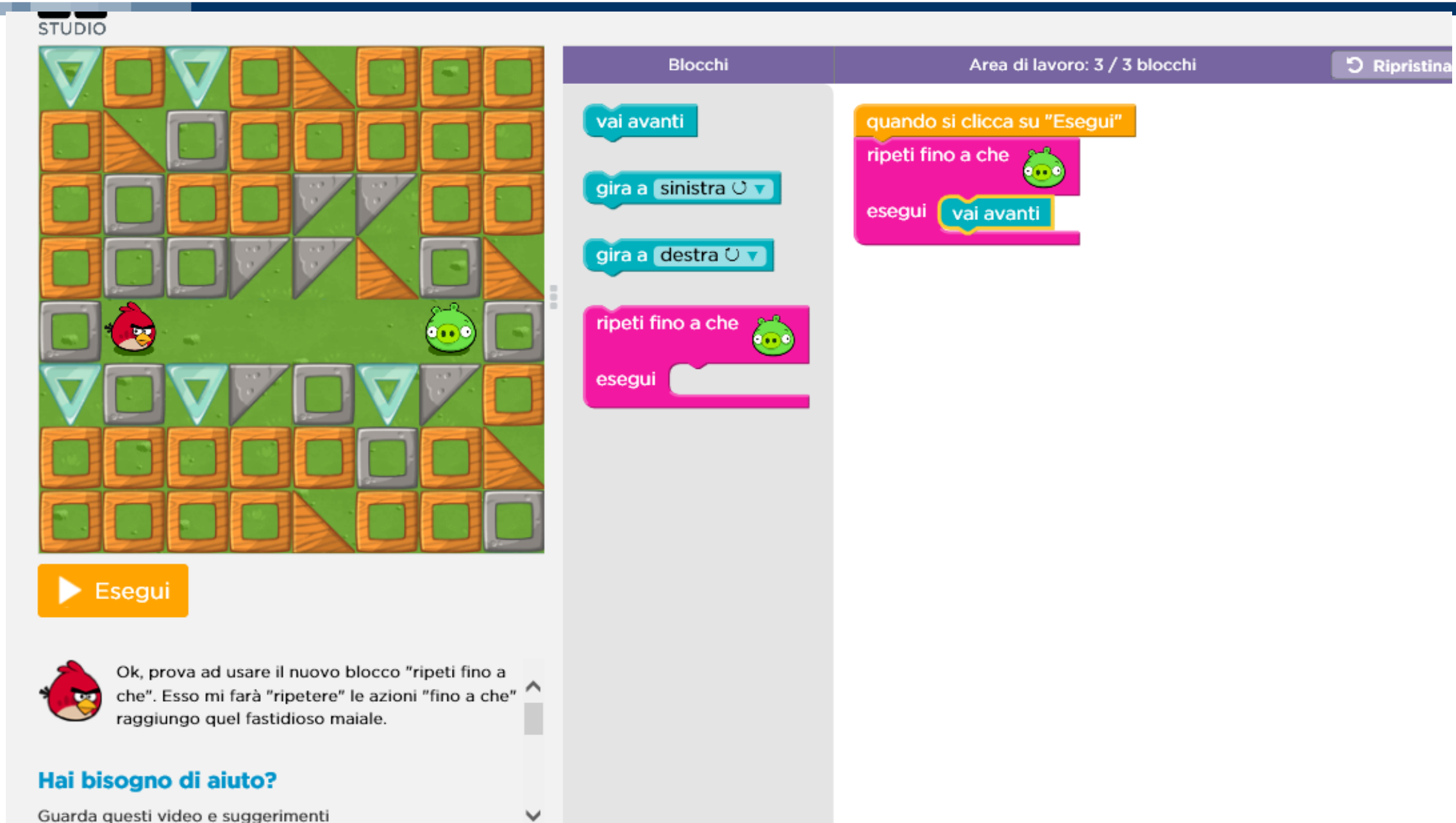


Se **`expression`** è vera eseguo il ramo di istruzioni contenente **`instrA;`** e **`instrB;`** (corpo del ciclo)

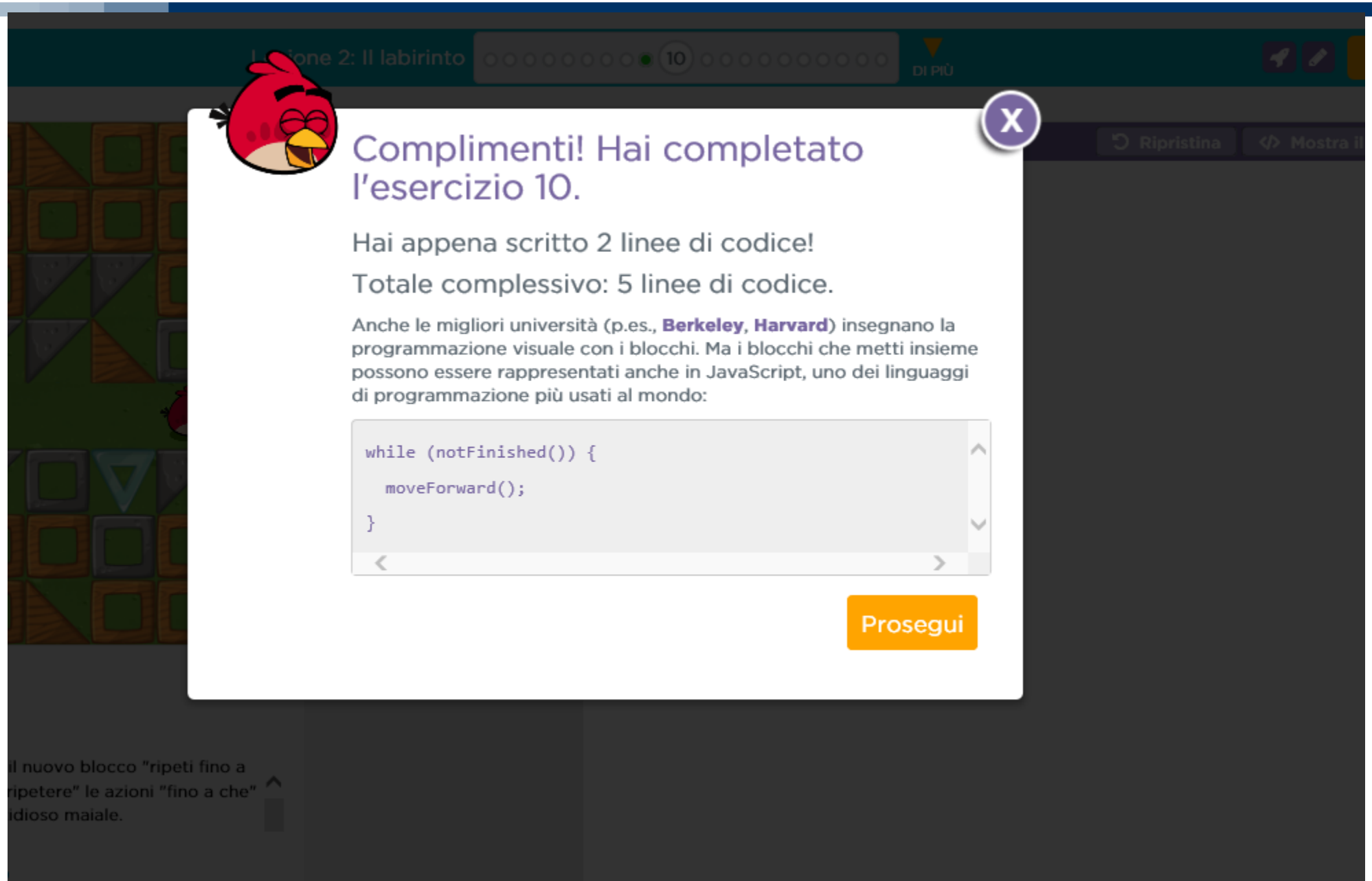
Al termine valuta nuovamente **`expression`**.

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

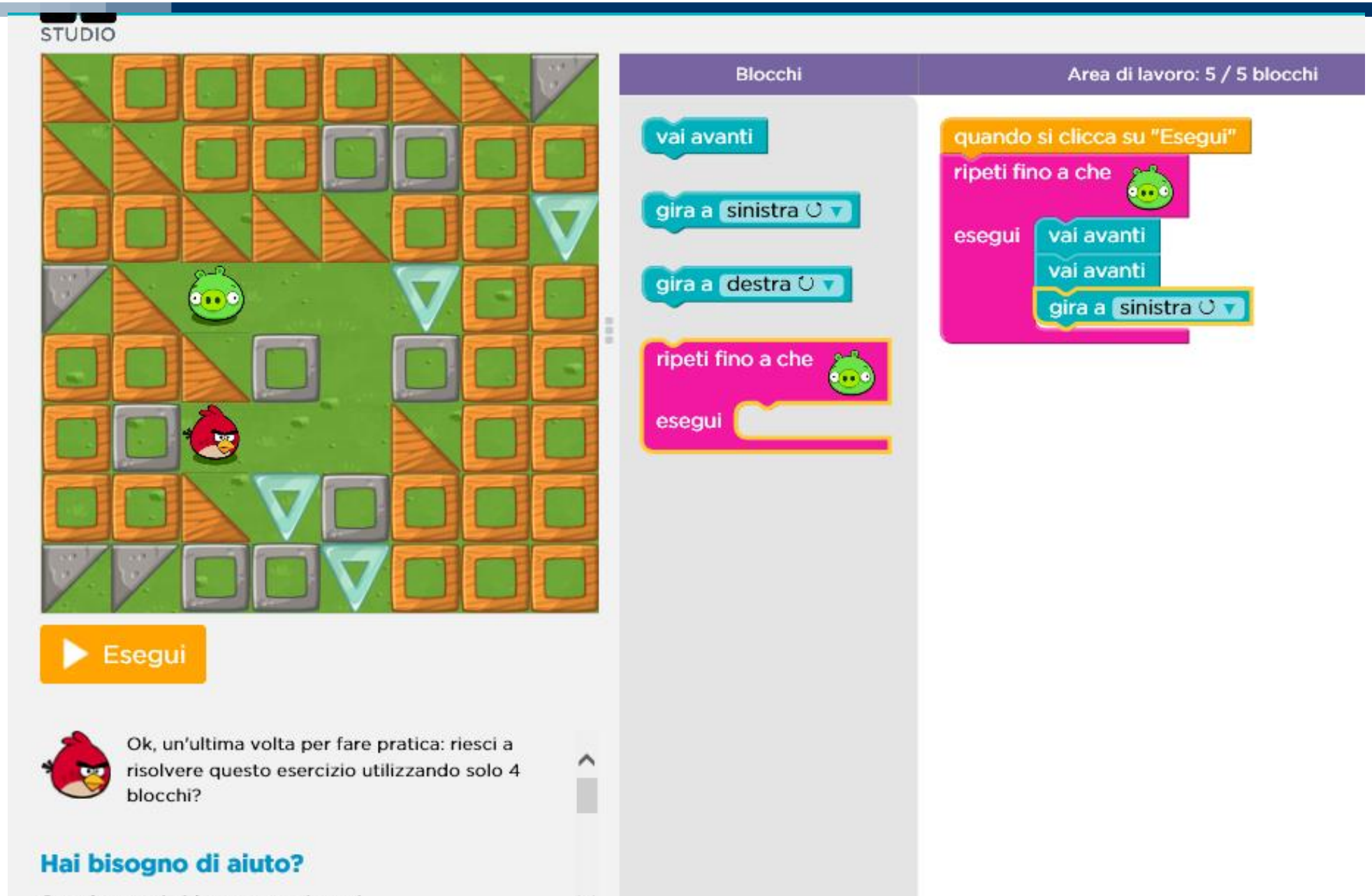
Il costrutto iterativo su code.org



Il costrutto iterativo su code.org



Il costrutto iterativo su code.org



Il costrutto iterativo su code.org

Il costrutto iterativo su code.org

Il costrutto iterative e condizionale su code.org

STUDIO

Esegui

Qui il nuovo blocco "se" viene usato per farmi decidere quando girare. Fai attenzione: adesso hai bisogno di inserire un solo blocco, ma capisci bene il procedimento in modo che puoi farlo tutto da solo la prossima volta.

Hai bisogno di aiuto?

Blocchi

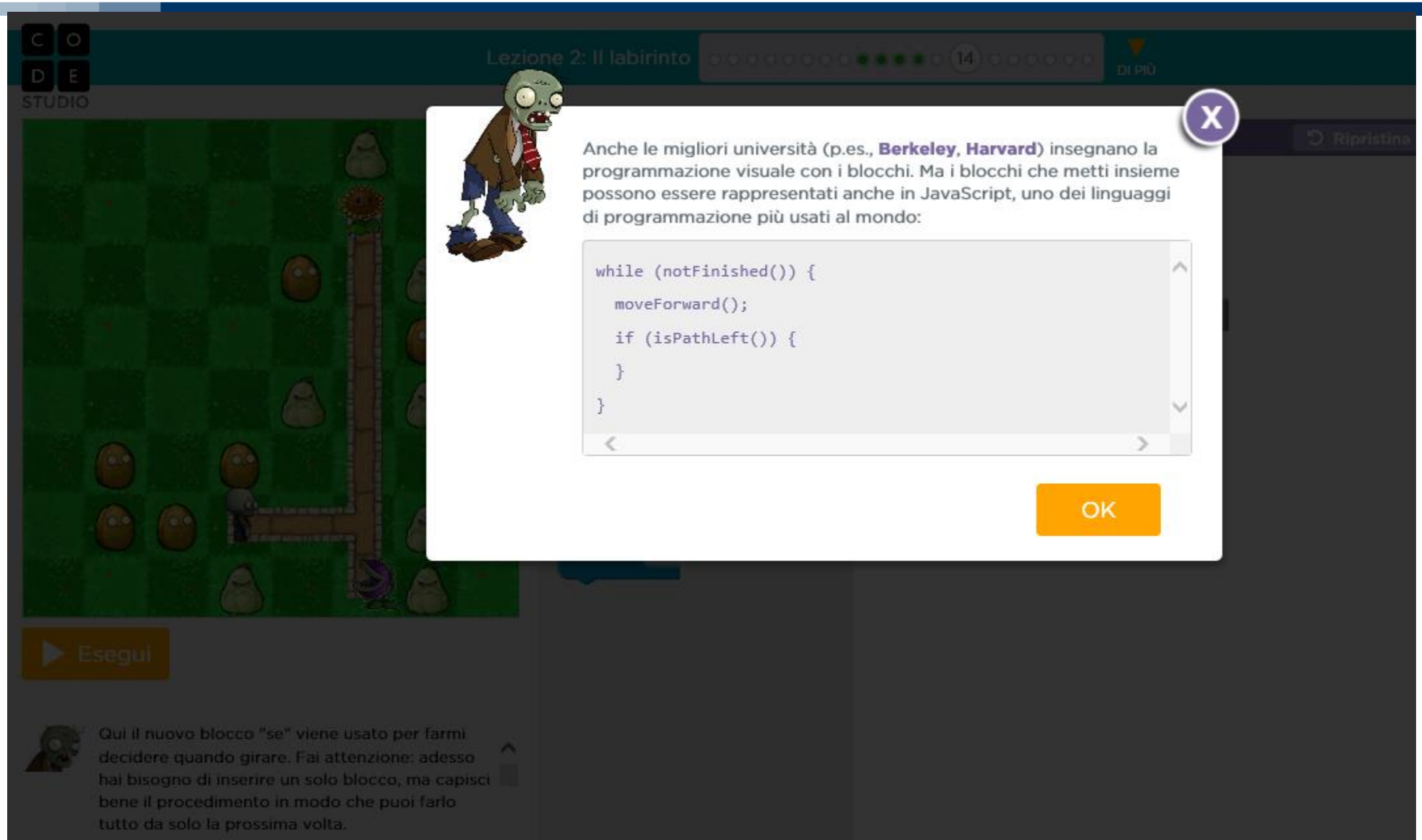
- vai avanti
- gira a sinistra
- gira a destra
- ripeti fino a che
  esegui
- se c'è strada a sinistra
  esegui

Area di lavoro: 4 / 5 blocchi

- quando si clicca su "Esegui"
  - ripeti fino a che
    esegui
    - vai avanti
    - se c'è strada a sinistra
      esegui

Ripete le azioni incluse, smettendo quando diventa

Il costrutto iterative su code.org



Esempi di Algoritmi

Il Pallottoliere

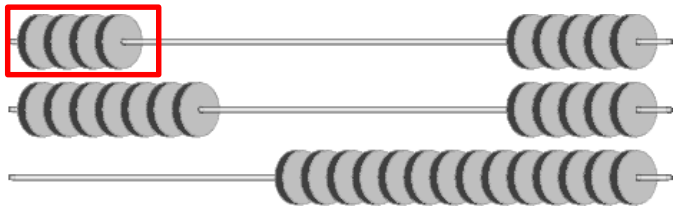
Supponiamo di avere un bambino che sa contare ma non sa fare operazioni aritmetiche.

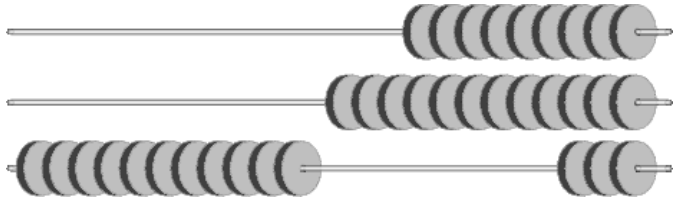
Scriviamo le istruzioni per utilizzare il pallottoliere (utilizzando un corstrutto iterativo)

Algoritmo per sommare col pallottoliere

Supponiamo che:

- Primo addendo sia riportato sulla prima riga a sinistra
- Secondo addendo sia sulla seconda riga a sinistra
- Risultato deve apparire nella terza riga (e che all'inizio questa abbia tutte le palline a destra).
- Operiamo con numeri «piccoli», non c'è bisogno del riporto

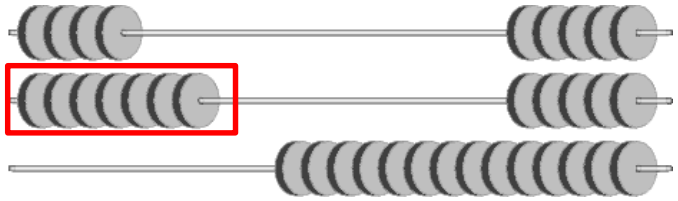


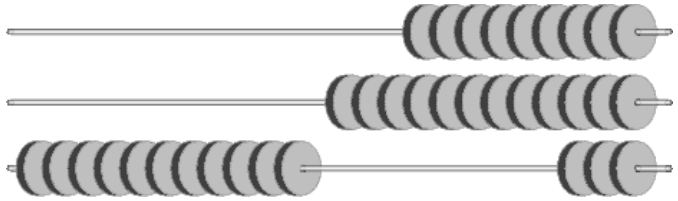


Algoritmo per sommare col pallottoliere

Supponiamo che:

- Primo addendo sia riportato sulla prima riga a sinistra
- Secondo addendo sia sulla seconda riga a sinistra
- Risultato deve apparire nella terza riga (e che all'inizio questa abbia tutte le palline a destra).
- Operiamo con numeri «piccoli», non c'è bisogno del riporto

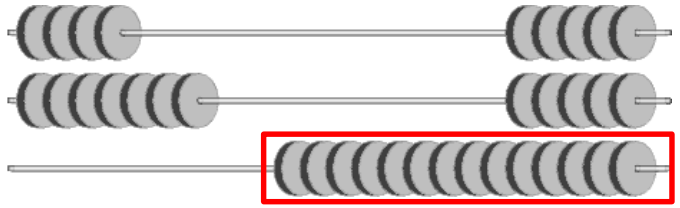


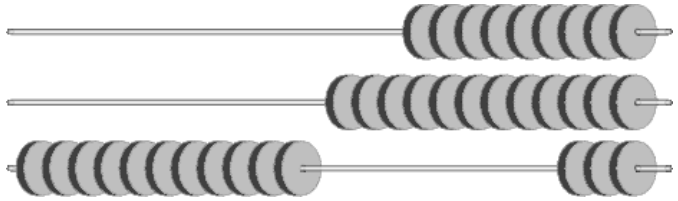


Algoritmo per sommare col pallottoliere

Supponiamo che:

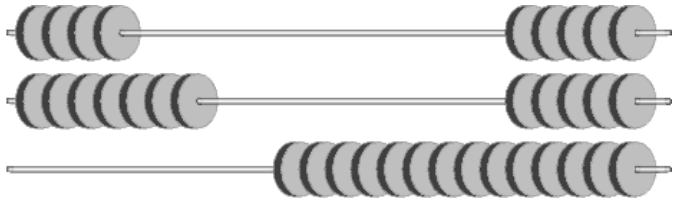
- Primo addendo sia riportato sulla prima riga a sinistra
- Secondo addendo sia sulla seconda riga a sinistra
- Risultato deve apparire nella terza riga (e che all'inizio questa abbia tutte le palline a destra).
- Operiamo con numeri «piccoli», non c'è bisogno del riporto

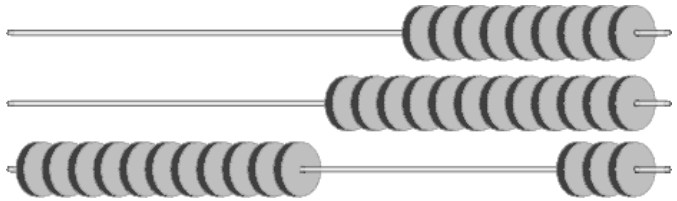




Algoritmo del pallottoliere

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 il bicchiere 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

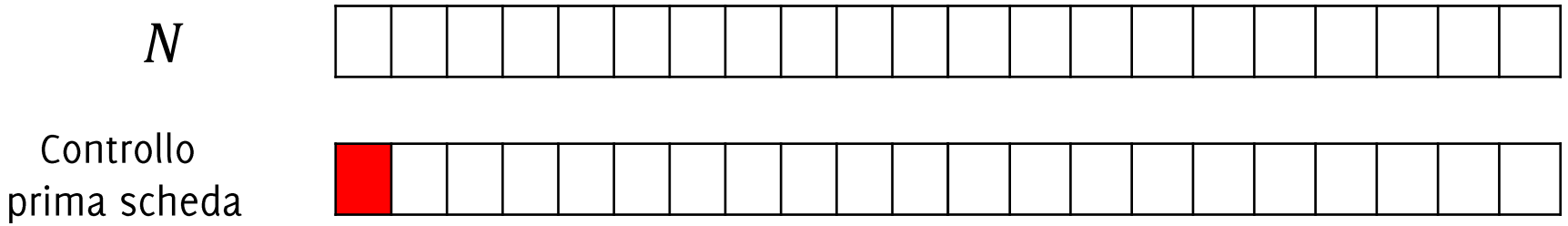
   altrimenti passa a scheda successiva
3. **Ripeti** istruzione **2**, **fino a conclusione o fino a raggiungere l'ultima scheda**
4. **se** trovata $\Rightarrow$ ricerca conclusa con successo
   - **altrimenti** $\Rightarrow$ ricerca conclusa con insuccesso

Algoritmo: Ricerca

Assumendo che l'archivio abbia N schede:

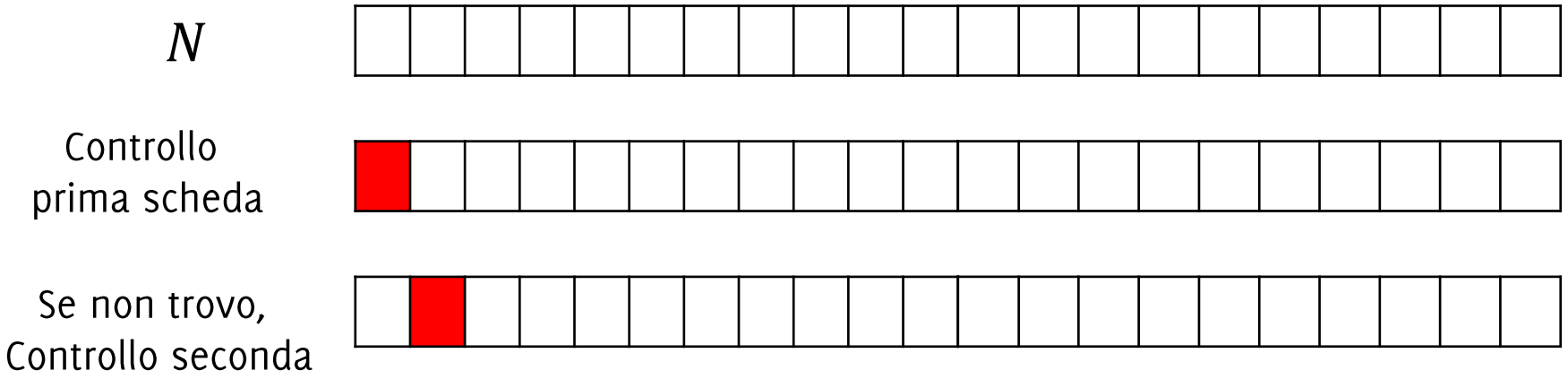
Algoritmo: Ricerca

Assumendo che l'archivio abbia N schede:



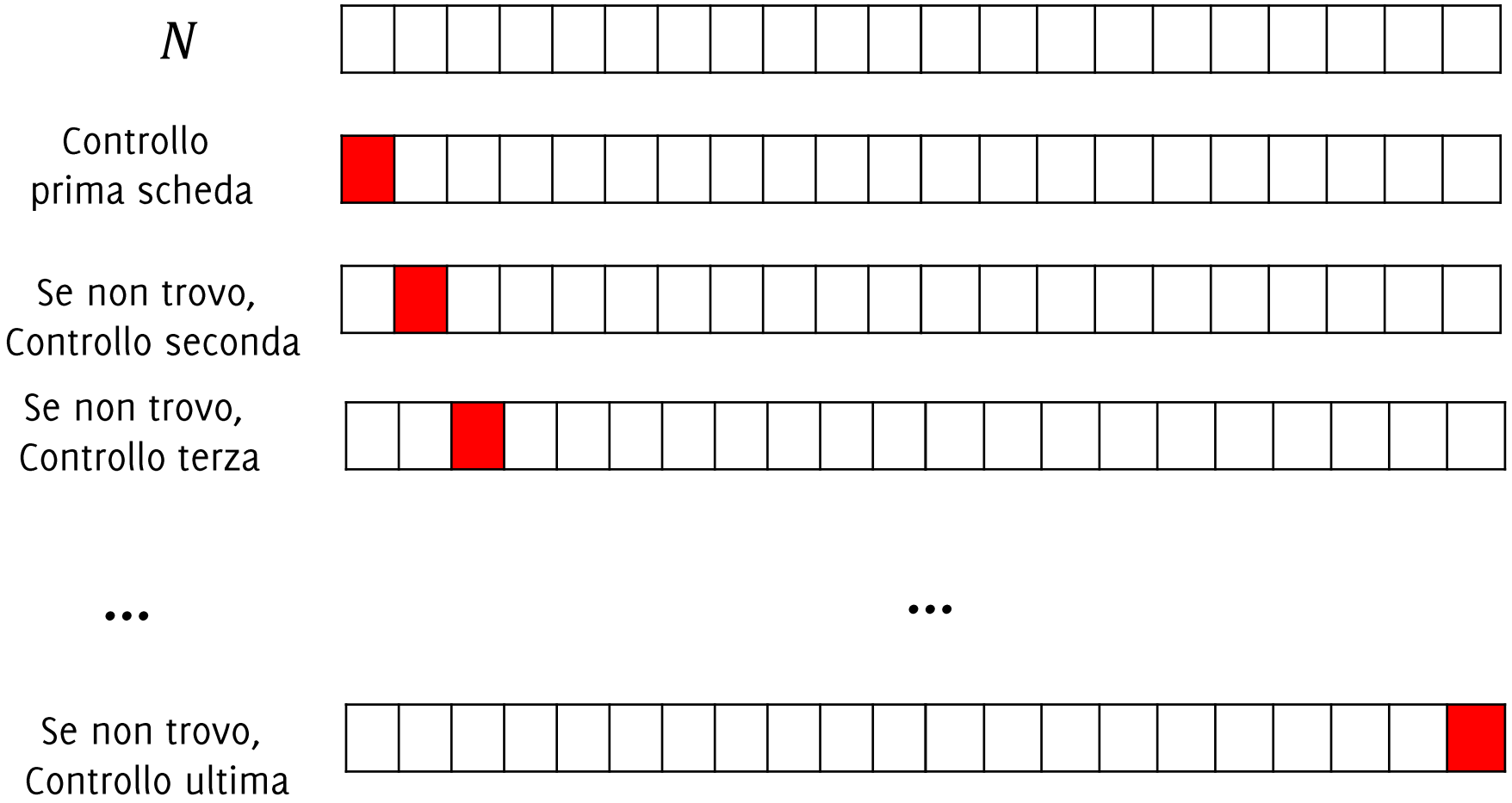
Algoritmo: Ricerca

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 ⇒ ricerca conclusa con successo
   - **altrimenti** ⇒ 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.

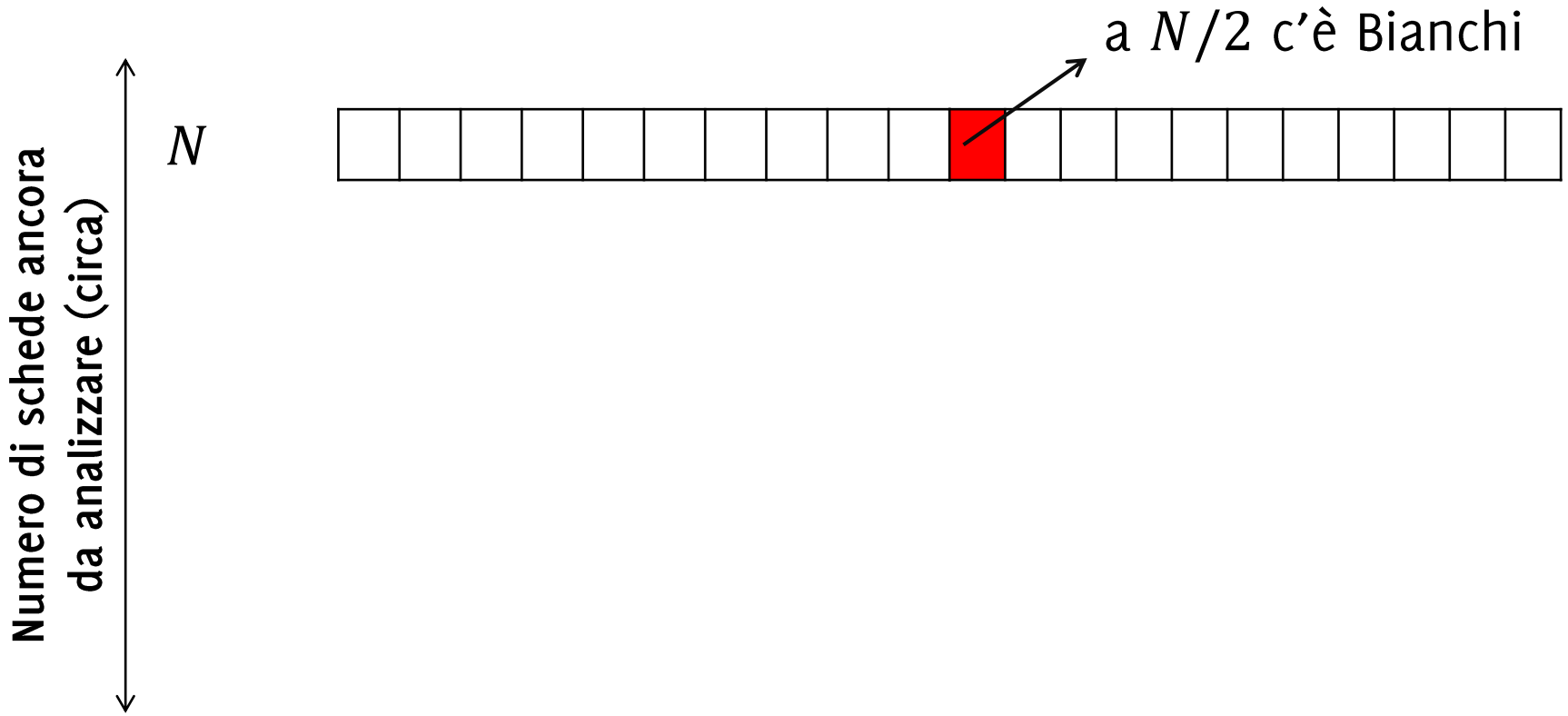
altrimenti,

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).

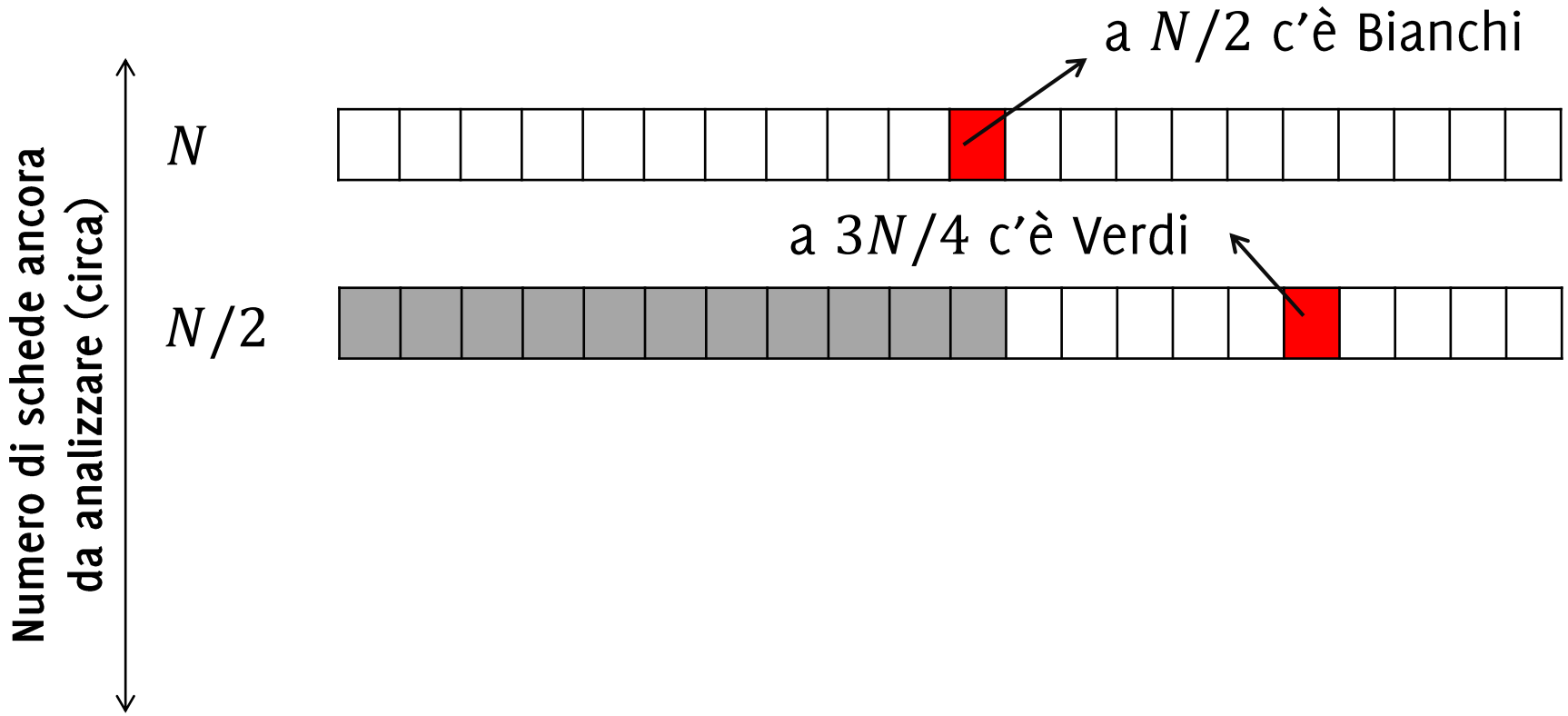
Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



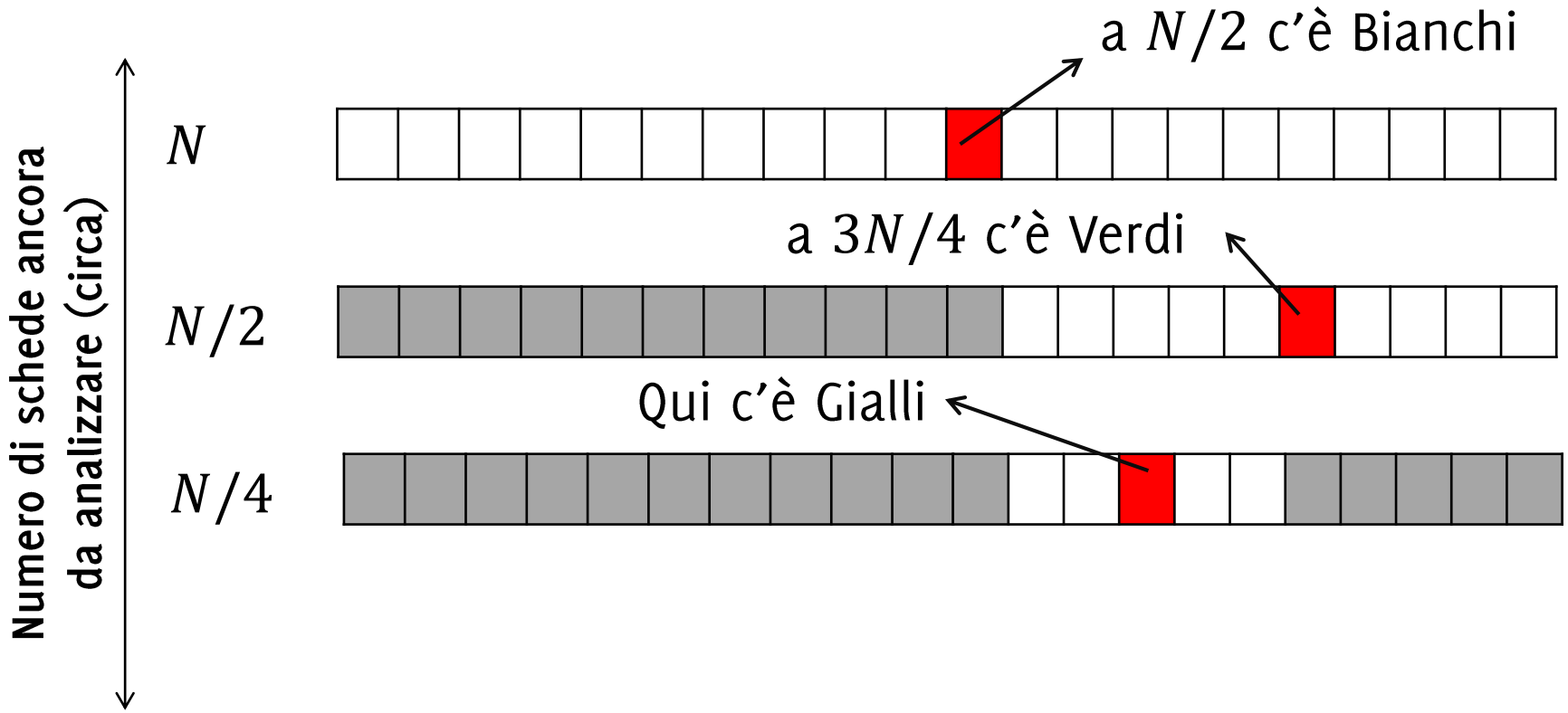
Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



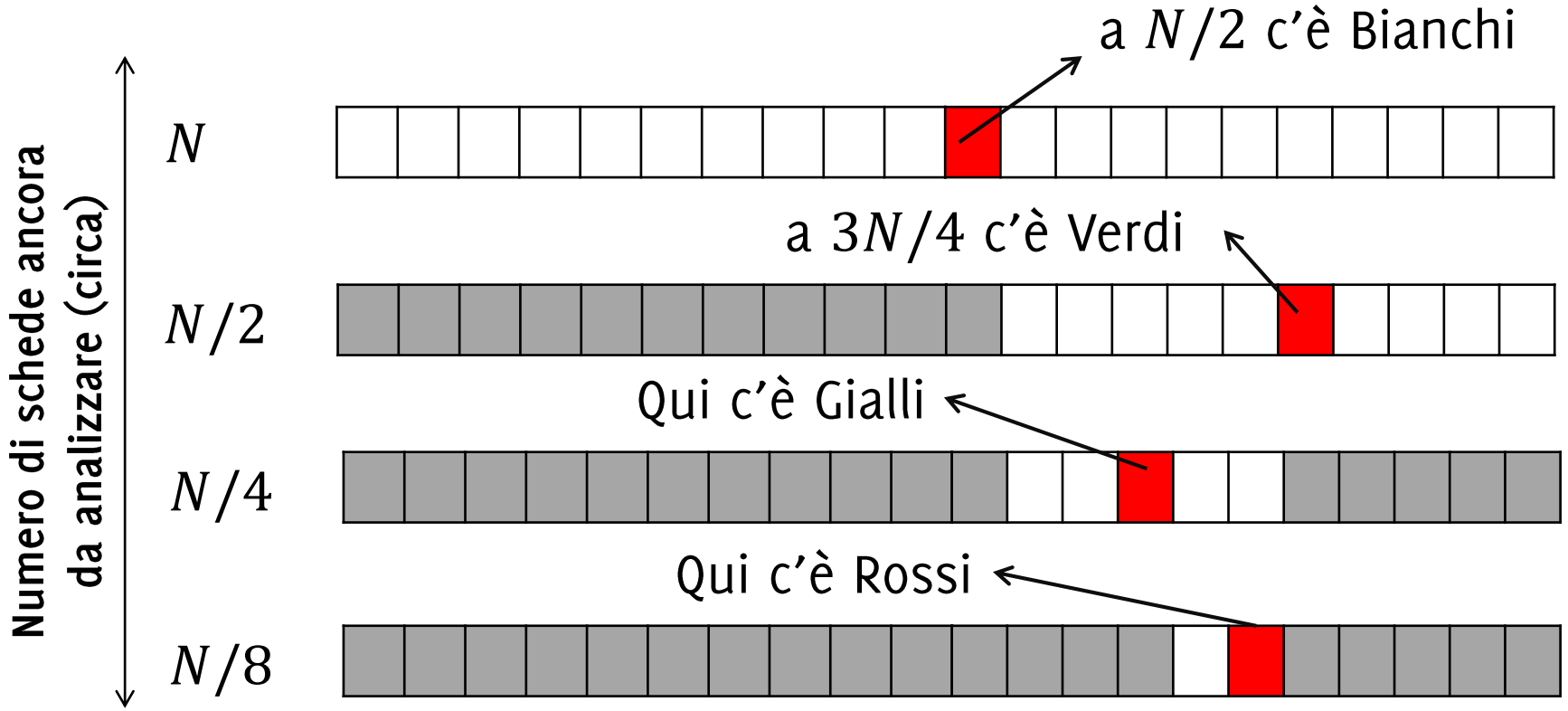
Algoritmo: Ricerca tra elementi ordinati

es: Ricerca dell'autore ROSSI



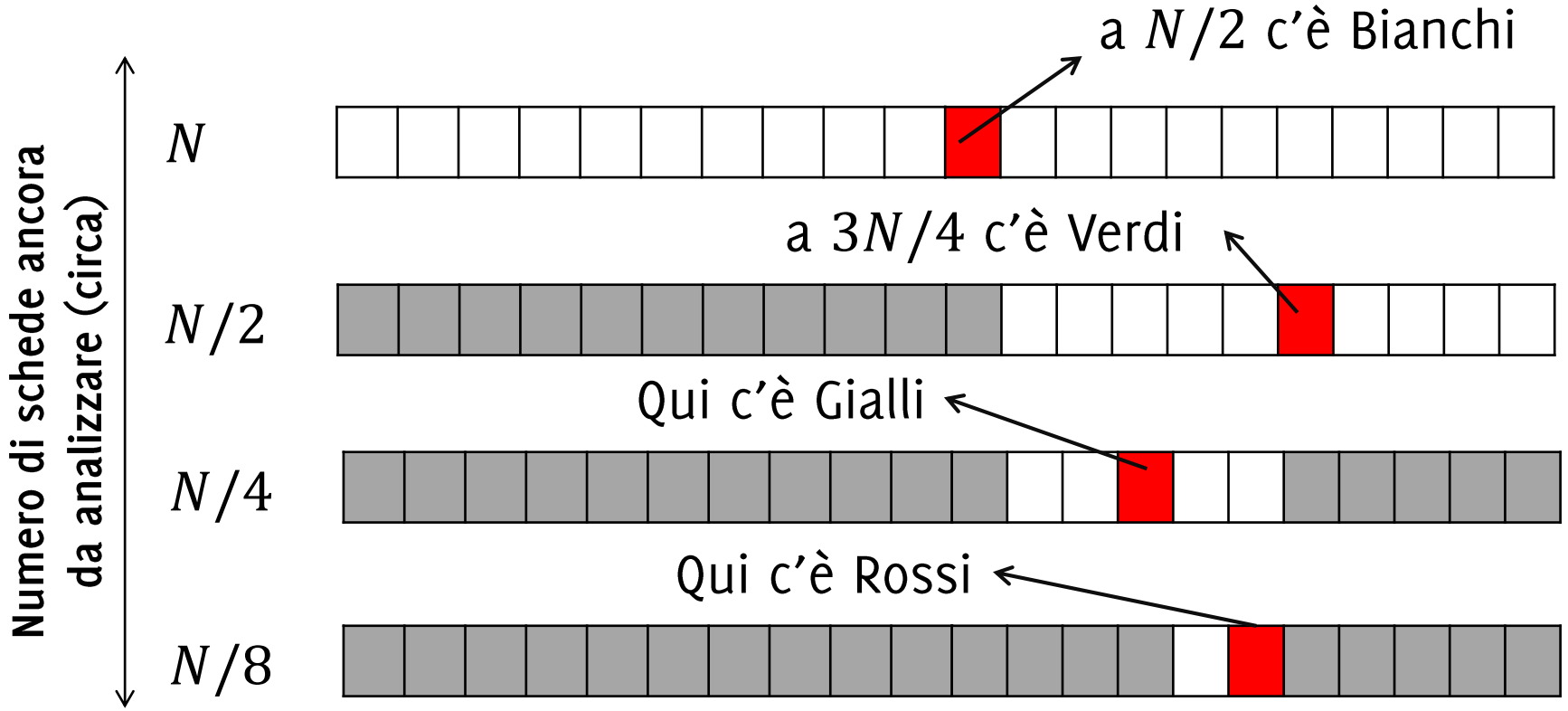
Algoritmo: Ricerca tra elementi ordinati

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

Intermezzo: Cosa fa questo algoritmo?

1. Alzatevi tutti in piedi
2. Ognuno di voi vale 1
3. **Ripeti:** Ciascuno cerca un compagno/a ancora in piedi
4. **Se** non avete trovato un compagno, il vostro valore non cambia e dovete restare in piedi
5. **Altrimenti** ogni coppia somma i loro valori, il risultato è il nuovo valore di ciascuno
6. Uno dei due si deve sedere e l'altro deve restare in piedi
7. Ricominciate dal punto 3, **finchè** non resta in piedi una sola persona in tutta la stanza
8. Il valore dell'ultima persona rimasta in piedi è

Intermezzo: Cosa fa questo algoritmo?

1. Alzatevi tutti in piedi
2. Ognuno di voi vale 1
3. **Ripeti:** Ciascuno cerca un compagno/a ancora in piedi
4. **Se** non avete trovato un compagno, il vostro valore non cambia e dovete restare in piedi
5. **Altrimenti** ogni coppia somma i loro valori, il risultato è il nuovo valore di ciascuno
6. Uno dei due si deve sedere e l'altro deve restare in piedi
7. Ricominciate dal punto 3, **finchè** non resta in piedi una sola persona in tutta la stanza
8. Il valore dell'ultima persona rimasta in piedi è il numero di persone presenti nella stanza

I Programmi

- Dall'algoritmo al programma

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 (o almeno ragionevole)

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

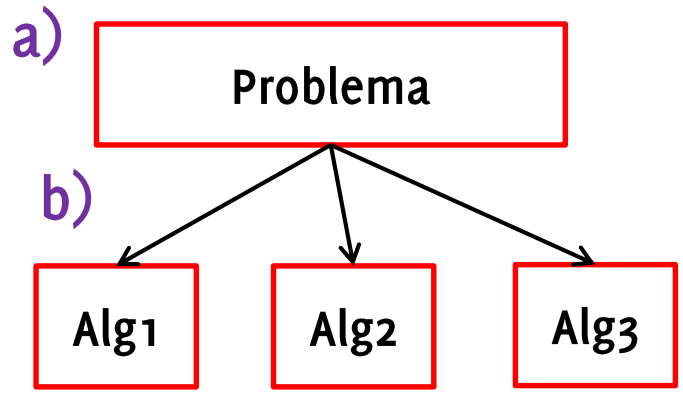
Riepilogando: Come Procedere

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

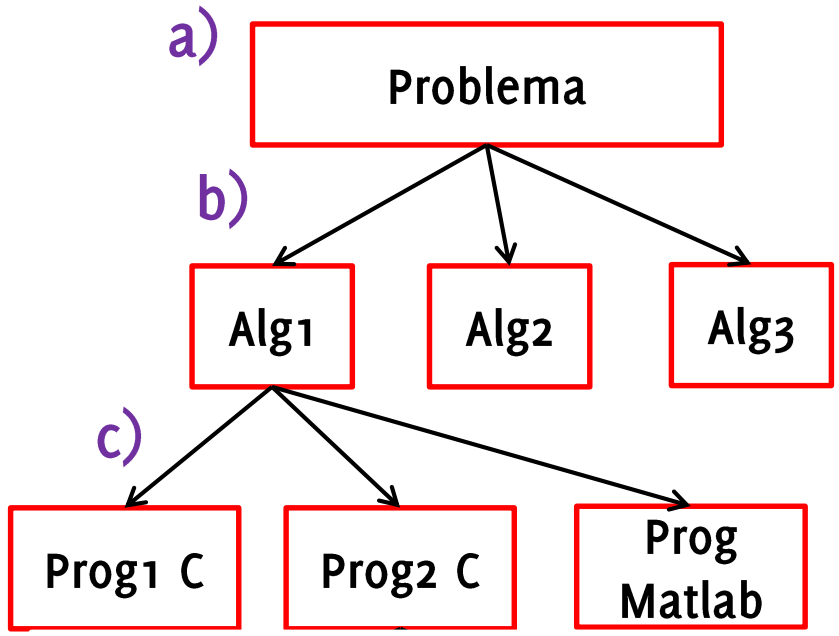


Riepilogando: Come Procedere

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,



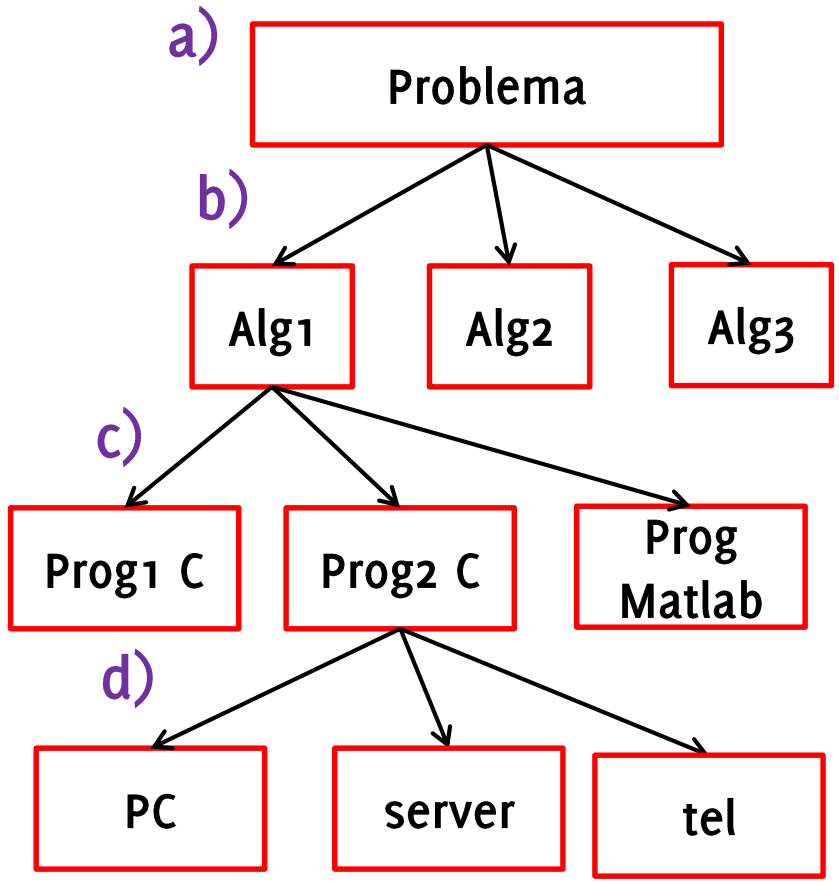
Riepilogando: Come Procedere

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 (nel vostro caso l'interprete Matlab)