



**INFORMATIK-BIBER SCHWEIZ  
 CASTOR INFORMATIQUE SUISSE  
 CASTORO INFORMATICO SVIZZERA**

**Quesiti e soluzioni 2022**

**11<sup>o</sup> al 13<sup>o</sup> anno scolastico**

<https://www.castoro-informatico.ch/>

A cura di:

Susanne Datzko, Nora A. Escherle, Masiar Babazadeh,  
 Christian Giang, Jean-Philippe Pellet

010100110101011001001001  
 010000010010110101010011  
 010100110100100101000101  
 001011010101001101010011  
 010010010100100100100001

**SS! I**

[www.svia-ssie-ssii.ch](http://www.svia-ssie-ssii.ch)  
 schweizerischerverein für informatik in  
 1erausbildung // société suisse pour l'infor  
 matique dans l'enseignement // società sviz  
 zera per l'informatica nell'insegnamento





# Hanno collaborato al Castoro Informatico 2022

Masiar Babazadeh, Susanne Datzko, Jean-Philippe Pellet, Giovanni Serafini, Bernadette Spieler

Capo progetto: Nora A. Escherle

Un particolare ringraziamento per il lavoro sui quesiti del concorso Svizzero va a:

Juraj Hromkovič, Christian Datzko, Jens Gallenbacher, Regula Lacher: ETH Zurich, Ausbildungs- und Beratungszentrum für Informatikunterricht

Tobias Berner: Pädagogische Hochschule Zürich

Waël Almoman: Collège Voltaire

La scelta dei quesiti è stata svolta in collaborazione con gli organizzatori dei concorsi in Germania, Austria, Ungheria, Slovacchia e Lituania. Ringraziamo specialmente:

Valentina Dagienė, Tomas Šiaulyš, Vaidotas Kinčius: Bebras.org

Wolfgang Pohl, Hannes Endreß, Ulrich Kiesmüller, Kirsten Schlüter, Michael Weigend: Bundesweite Informatikwettbewerbe (BWINF), Germania

Wilfried Baumann, Liam Baumann, Anoki Eischer, Thomas Galler, Benjamin Hirsch, Martin Kandlhofer, Katharina Resch-Schobel: Österreichische Computer Gesellschaft

Gerald Futschek, Florentina Voboril: Technische Universität Wien

Zsuzsa Pluhár: ELTE Informatikai Kar, Ungheria

Michal Winzcer: Comenius University, Slovacchia

La versione online del concorso è stata creata su [cuttle.org](http://cuttle.org). Ringraziamo per la buona collaborazione: Eljakim Schrijvers, Justina Dauksaite, Dave Oostendorp, Alieke Stijf, Kyra Willekes, Jo-Ann Bolten: [cuttle.org](http://cuttle.org), Olanda

Chris Roffey: UK Bebras Administrator, Regno Unito

Per il supporto durante le settimane del concorso ringraziamo:

Hanspeter Erni: Direttore scuola media di Rickenbach

Christoph Frei: Chragokyberneticks (Logo Informatik-Biber Schweiz)

Dr. Andrea Leu, Maggie Winter, Lena Frölich: Senarclens Leu + Partner AG

L'edizione dei quesiti in lingua tedesca è stata utilizzata anche in Germania e in Austria.

La traduzione francese è stata curata da Elsa Pellet mentre quella italiana da Christian Giang.



**INFORMATIK-BIBER SCHWEIZ**  
**CASTOR INFORMATIQUE SUISSE**  
**CASTORO INFORMATICO SVIZZERA**

Il Castoro Informatico 2022 è stato organizzato dalla Società Svizzera per l'Informatica nell'Insegnamento (SSII) con il sostegno determinante della fondazione Hasler. Gli sponsor del concorso sono l'Ufficio per l'economia e il lavoro del Cantone di Zurigo e UBS.

Questo quaderno è stato creato il 18 agosto 2025 con il sistema per la preparazione di testi  $\text{\LaTeX}$ . Ringraziamo Christian Datzko per lo sviluppo del sistema di generazione dei testi che ha permesso di generare le 36 versioni di questa brochure (divise per lingua e livello scolastico). Il sistema è stato riprogrammato basandosi sul sistema precedente, sviluppato nel 2014 assieme a Ivo Blöchliger. Ringraziamo Jean-Philippe Pellet per lo sviluppo del sistema `bebras`, utilizzato dal 2020 per la conversione dei documenti sorgente dai formati Markdown e YAML.

Nota: Tutti i link sono stati verificati l'01.12.2022.



I quesiti sono distribuiti con Licenza Creative Commons Attribuzione – Non commerciale – Condividi allo stesso modo 4.0 Internazionale. Gli autori sono elencati a pagina 57.



## Premessa

Il concorso del «Castoro Informatico», presente già da diversi anni in molti paesi europei, ha l'obiettivo di destare l'interesse per l'informatica nei bambini e nei ragazzi. In Svizzera il concorso è organizzato in tedesco, francese e italiano dalla Società Svizzera per l'Informatica nell'Insegnamento (SSII), con il sostegno della fondazione Hasler.

Il Castoro Informatico è il partner svizzero del Concorso «Bebras International Contest on Informatics and Computer Fluency» (<https://www.bebas.org/>), situato in Lituania.

Il concorso si è tenuto per la prima volta in Svizzera nel 2010. Nel 2012 l'offerta è stata ampliata con la categoria del «Piccolo Castoro» (3<sup>o</sup> e 4<sup>o</sup> anno scolastico).

Il Castoro Informatico incoraggia gli alunni ad approfondire la conoscenza dell'informatica: esso vuole destare interesse per la materia e contribuire a eliminare le paure che sorgono nei suoi confronti. Il concorso non richiede alcuna conoscenza informatica pregressa, se non la capacità di «navigare» in internet poiché viene svolto online. Per rispondere alle domande sono necessari sia un pensiero logico e strutturato che la fantasia. I quesiti sono pensati in modo da incoraggiare l'utilizzo dell'informatica anche al di fuori del concorso.

Nel 2022 il Castoro Informatico della Svizzera è stato proposto a cinque differenti categorie d'età, suddivise in base all'anno scolastico:

- 3<sup>o</sup> e 4<sup>o</sup> anno scolastico («Piccolo Castoro»)
- 5<sup>o</sup> e 6<sup>o</sup> anno scolastico
- 7<sup>o</sup> e 8<sup>o</sup> anno scolastico
- 9<sup>o</sup> e 10<sup>o</sup> anno scolastico
- 11<sup>o</sup> al 13<sup>o</sup> anno scolastico

Ogni categoria aveva quesiti classificati in tre livelli di difficoltà: facile, medio e difficile. Alla categoria del 3<sup>o</sup> e 4<sup>o</sup> anno scolastico sono stati assegnati 9 quesiti da risolvere, di cui 3 facili, 3 medi e 3 difficili. Alla categoria del 5<sup>o</sup> e 6<sup>o</sup> anno scolastico sono stati assegnati 12 quesiti, suddivisi in 4 facili, 4 medi e 4 difficili. Ogni altra categoria ha ricevuto invece 15 quesiti da risolvere, di cui 5 facili, 5 medi e 5 difficili.

Per ogni risposta corretta sono stati assegnati dei punti, mentre per ogni risposta sbagliata sono stati detratti. In caso di mancata risposta il punteggio è rimasto inalterato. Il numero di punti assegnati o detratti dipende dal grado di difficoltà del quesito:

	Facile	Medio	Difficile
Risposta corretta	6 punti	9 punti	12 punti
Risposta sbagliata	-2 punti	-3 punti	-4 punti

Il sistema internazionale utilizzato per l'assegnazione dei punti limita l'eventualità che il partecipante possa ottenere buoni risultati scegliendo le risposte in modo casuale.



Ogni partecipante inizia con un punteggio pari a 45 punti (risp., Piccolo Castoro: 27 punti, 5<sup>o</sup> e 6<sup>o</sup> anno scolastico: 36 punti).

Il punteggio massimo totalizzabile era dunque pari a 180 punti (risp., Piccolo castoro: 108 punti, 5<sup>o</sup> e 6<sup>o</sup> anno scolastico: 144 punti), mentre quello minimo era di 0 punti.

In molti quesiti le risposte possibili sono state distribuite sullo schermo con una sequenza casuale. Lo stesso quesito è stato proposto in più categorie d'età. Questi quesiti presentavano livelli di difficoltà diversi nei vari gruppi di età.

Alcuni quesiti sono indicati come «bonus» per determinate categorie di età: non contano nel totale dei punti, ma vengono utilizzati come spareggio per punteggi identici in caso di qualificazione agli eventuali turni successivi.

### **Per ulteriori informazioni:**

SVIA-SSIE-SSII Società Svizzera per l'Informatica nell'Insegnamento  
Castoro Informatico  
Masiar Babazadeh

<https://www.castoro-informatico.ch/it/kontaktieren/>  
<https://www.castoro-informatico.ch/>



# Indice

Hanno collaborato al Castoro Informatico 2022 . . . . .	i
Premessa . . . . .	iii
Indice . . . . .	v
1. La posta robotizzata . . . . .	1
2. Sequenze di dati . . . . .	5
3. Capannone rotante . . . . .	9
4. Serata cinematografica . . . . .	13
5. Tris . . . . .	17
6. Pietre preziose . . . . .	21
7. Ciottoli e conchiglie . . . . .	23
8. Maria alla caccia del tesoro . . . . .	27
9. Incarto di cioccolatini . . . . .	31
10. Labirinto . . . . .	35
11. Virus . . . . .	39
12. Colorazione del pavimento . . . . .	43
13. Mosaico . . . . .	47
14. Oggetti magici . . . . .	51
15. Campionato dei Castori . . . . .	55
A. Autori dei quesiti . . . . .	57
B. Sponsoring: concorso 2022 . . . . .	58
C. Ulteriori offerte . . . . .	59





# 1. La posta robotizzata

Il robot Tina consegna la posta. Tina utilizza una mappa suddivisa in campi. Tina si sposta lungo la strada verso una strada adiacente a sinistra, a destra o davanti (cioè non in diagonale).

Tina ha tre sensori per la navigazione. Non appena Tina entra in una strada (e prima che Tina possa girarsi), i sensori rilevano ciò che si trova a sinistra, a destra e di fronte a Tina.

La tabella documenta ciò che i sensori di Tina hanno rilevato in ogni casella del suo percorso. Tina inizia sulla casella , in direzione della freccia.

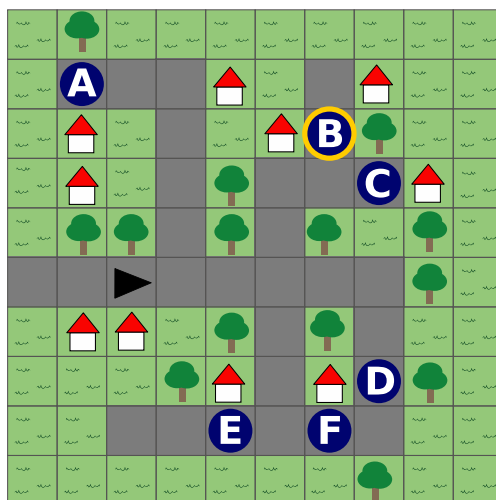
	sinistra	davanti	destra

























Quale dei punti blu scuro Tina raggiungerà alla fine del suo percorso?










## Soluzione

La risposta corretta è il punto B.

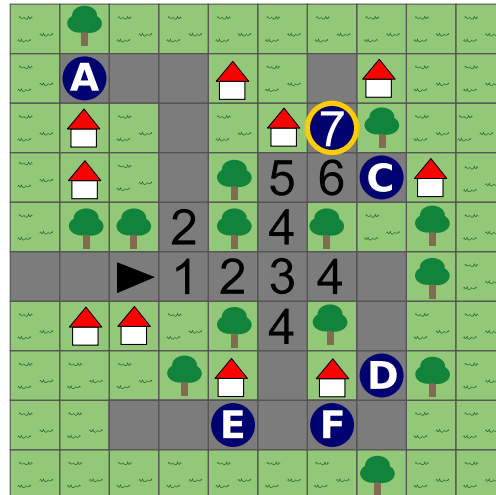


Passo	sinistra	davanti	destra
			
1			
2			
3			
4			
5			
6			
7			

In questo caso è sufficiente concentrarsi sui sei punti di destinazione e vedere se le indicazioni del sensore del passaggio 7 «  » possono essere adatte. In questo modo è possibile escludere C, E e F. Le specifiche del sensore del passo 6 sono «  », quindi è possibile escludere A e D.

In alternativa, si può provare a seguire il percorso documentato nella tabella. Il percorso verso il punto B è l'unico che corrisponde.

Se si traccia il percorso di Tina utilizzando le informazioni dei sensori, non è sempre possibile decidere immediatamente dove Tina si è spostata. Nel passo 4, Tina vedeva gli alberi a sinistra e a destra, indipendentemente dalle tre direzioni in cui si muoveva. In questa situazione, è necessario prendere in considerazione anche le informazioni del sensore dopo il movimento successivo per poter determinare chiaramente il punto 4.



## Questa è l'informatica!

In questo compito incontriamo il *robot* Tina. I robot sono computer appositamente attrezzati che raccolgono informazioni dall'ambiente circostante con l'aiuto di *sensori*, le elaborano automaticamente (cioè con un programma) e, in base al risultato, eseguono autonomamente un'azione nel loro ambiente attraverso i cosiddetti *attuatori*. I sensori di Tina rilevano innanzitutto il contenuto delle caselle sinistra, davanti e destra. Nello specifico, potremmo immaginare che i sensori scattino foto e che dall'analisi automatizzata di queste immagini vengano estratti dati geometrici che il computer può assegnare a una casa, un albero o una strada. Il corpo di Tina, cioè gli *attuatori*, potrebbero essere controllati per evitare campi con alberi o una casa.

Le auto a guida autonoma sono esempi famosi di questi robot. Sono dotati di numerosi sensori che non solo misurano la velocità o la posizione corrente, ma anche la distanza dal ciglio della strada e rilevano gli oggetti presenti sulla strada o a bordo strada e molto altro ancora. Queste informazioni vengono elaborate da programmi a volte molto complessi che possono, ad esempio, riconoscere i bambini che potrebbero attraversare la strada e distinguerli da un cartello stradale. In molti di questi scenari, il cosiddetto *apprendimento automatico* è la tecnologia chiave. Nel caso delle auto a guida autonoma, i computer imparano, sulla base di numerosi esempi, a distinguere i bambini dai segnali stradali. Gli *attuatori* sono quindi, ad esempio, i freni, che vengono attivati in modo indipendente o senza l'intervento umano.

## Parole chiave e siti web

- Robot: <https://it.wikipedia.org/wiki/Robot>
- Sensore: <https://it.wikipedia.org/wiki/Sensore>
- Attuatore: <https://it.wikipedia.org/wiki/Attuatore>
- Apprendimento automatico: [https://it.wikipedia.org/wiki/Apprendimento\\_automatico](https://it.wikipedia.org/wiki/Apprendimento_automatico)





## 2. Sequenze di dati

Qui possiamo vedere una sequenza di numeri di nome X. Nelle posizioni da 1 a 5 della sequenza X si trovano i seguenti numeri: 5, 3, 2, 4, 1.

1 2 3 4 5  
X 5 3 2 4 1

Descriviamo il numero in una certa posizione mettendo tra parentesi il nome e la posizione. Un esempio: descriviamo il numero in posizione 2 della sequenza X in questo modo: (X 2). Attualmente, (X 2) = 3.

Un numero nella sequenza così descritta può essere esso stesso una posizione. Ad esempio, (X (X 2)) = (Xv3) = 2.

Ecco altre tre sequenze: A, B e C.

A 3 2 4 1 5  
B 5 4 1 3 2  
C 2 5 4 3 1

Quale numero descriviamo in questo modo: (A (B (C 3))) ?

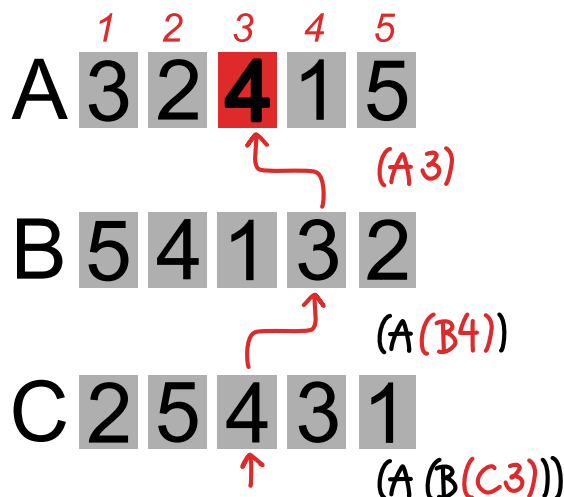
- A) 1
- B) 2
- C) 3
- D) 4
- E) 5



## Soluzione

La risposta corretta è D) 4.

La descrizione (A (B (C 3))) dice: il numero descritto si trova nella sequenza A alla posizione (B (C 3)); la posizione si trova quindi nella sequenza B alla posizione (C 3); e questa posizione si trova a sua volta nella sequenza C alla posizione 3. Complicato!



È più facile se valutiamo la descrizione «dall'interno verso l'esterno», come con un'espressione aritmetica - e come è già stato dimostrato nell'esempio del compito:  $(A (B (C 3))) = (A (B 4)) = (A 3) = 4$

## Questa è l'informatica!

Non molto tempo fa si parlava di *elaborazione dei dati* quando si parlava dell'uso dei computer. Giustamente, perché i computer elaborano tutti i tipi di dati, come numeri, testi, immagini, suoni, ecc. La maggior parte dei dati interessanti memorizzati nei computer sono di natura complessa e hanno una struttura: le temperature misurate nel corso della giornata in una stazione meteorologica, ad esempio, possono essere memorizzate come una sequenza di coppie, ciascuna composta dall'ora della misurazione e dalla temperatura misurata. Quindi c'è una struttura a coppie e una struttura a sequenze.

I dati possono avere un'ampia varietà di strutture e per questo gli informatici hanno sviluppato un'ampia varietà di cosiddette *strutture di dati* per memorizzare i dati in modo intelligente e (altrettanto importante) per accedere ai dati in modo efficiente. Una semplice struttura di dati è l'*array*, che svolge il ruolo principale in questo compito. Un array può memorizzare un numero fisso di dati (compresi i numeri) in posizioni successive. A causa delle posizioni, i dati nell'array hanno una struttura ordinata - un array sarebbe quindi adatto per le coppie tempo/temperatura menzionate sopra. A causa della loro dimensione fissa, gli array appartengono alle strutture dati *statiche* dell'informatica. Per le sequenze di dati, esistono anche strutture di dati *dinamiche* come le liste, la cui dimensione può cambiare a seconda delle necessità.



Statico o dinamico: se una struttura di dati di una sequenza contiene numeri, questi numeri possono anche indicare posizioni nella stessa o in un'altra sequenza. Questo viene spesso utilizzato in informatica per il cosiddetto indirizzamento indiretto: L'indirizzo o la posizione in una sequenza non è specificato direttamente come numero, ma indirettamente da un valore (numerico) di una sequenza, che a sua volta può essere indirizzata di nuovo indirettamente - e così via.



## Parole chiave e siti web

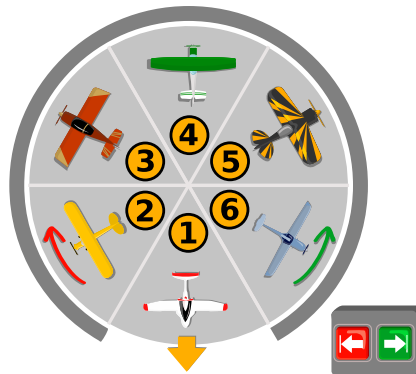
- Elaborazione dati: [https://it.wikipedia.org/wiki/Elaborazione\\_dati](https://it.wikipedia.org/wiki/Elaborazione_dati)
- Struttura dati: [https://it.wikipedia.org/wiki/Struttura\\_dati](https://it.wikipedia.org/wiki/Struttura_dati)
- Array: <https://it.wikipedia.org/wiki/Array>
- Metodo di indirizzamento: [https://it.wikipedia.org/wiki/Metodo\\_di\\_indirizzamento](https://it.wikipedia.org/wiki/Metodo_di_indirizzamento)






### 3. Capannone rotante

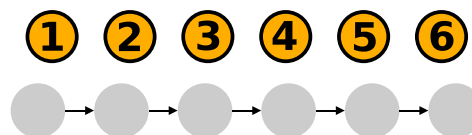
Nel campo di aviazione di Beavertown, sei aerei sono parcheggiati in un capannone. Sono su una piattaforma rotante, parcheggiati in sei posizioni diverse. All'esterno sono presenti due tasti freccia  . Con un solo tasto è possibile ruotare l'unità di rotazione esattamente di una posizione di parcheggio a sinistra o a destra.



Al mattino, quando i piloti ritirano i loro aerei, la posizione di parcheggio 1 è sempre sulla porta del capannone e l'aereo su di essa può uscire. Nel migliore dei casi, i tasti freccia devono essere premuti altre cinque volte, in modo che anche tutti gli altri aerei possano uscire. Ad esempio, se i piloti vogliono accedere alle posizioni di parcheggio nell'ordine 1, 6, 5, 4, 3, 2, è sufficiente premere cinque volte il tasto .

Ma qual è il caso peggiore? In quale ordine devono essere premuti più spesso i tasti?

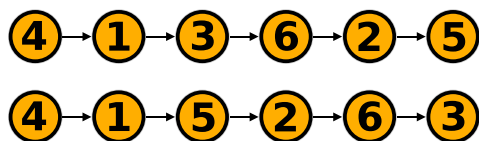
*Fornisci un esempio di un ordine di questo tipo.*





## Soluzione

Ci sono due risposte corrette:



Per trovare la soluzione, viene sempre selezionato l'aereo che si trova nella posizione di parcheggio con la distanza maggiore dalla porta del capannone.

« 4» significa che dopo aver premuto tre tasti l'aeromobile si parcheggerà alla posizione 4

4 1 3 6 2 5:



4 1 5 2 6 3:



In entrambi i casi, sono necessari un totale di 16 passi.

I passi non possono essere più di 16, perché solo all'inizio possono susseguirsi due passaggi con tre pressioni dei tasti freccia. Dopodiché, si possono alternare al massimo due e tre passi.

## Questa è l'informatica!

Il capannone rotante ha il vantaggio di poter parcheggiare gli aerei in modo molto poco ingombrante. Tuttavia, la raccolta di solito richiede più tempo rispetto a un normale capannone.

L'*efficienza* di una procedura è un argomento centrale in informatica perché è un importante criterio di valutazione per gli *algoritmi*. Molto spesso l'efficienza riguarda il tempo di esecuzione, ma non è sempre così. Nella definizione generale di efficienza di un algoritmo, si tratta di tutte le risorse necessarie, quindi anche, ad esempio, della dimensione della memoria necessaria (*efficienza della memoria*).

Come nell'esempio del capannone, il risparmio di una risorsa porta a un aumento della domanda di un'altra risorsa. Dipende dal contesto specifico e dalla disponibilità delle risorse a quale risorsa viene data maggiore importanza.

Ad esempio, *bubblesort* e *timsort* sono entrambi algoritmi per ordinare un elenco di elementi. Bubblesort ordina l'elenco in un tempo proporzionale al numero di elementi al quadrato ( $\mathcal{O}(n^2)$ ), ma richiede poca memoria aggiuntiva, costante rispetto alla lunghezza dell'elenco.

Timsort ordina molto più velocemente di bubblesort ( $\mathcal{O}(n \log n)$ ), ma ha un requisito di spazio che aumenta linearmente con la dimensione dell'elenco. Se per una particolare applicazione è necessario



ordinare ad alta velocità elenchi di grandi dimensioni, Timsort è la scelta migliore; se invece è più importante ridurre al minimo i requisiti di memoria dell'ordinamento, Bubblesort è la scelta migliore.

## Parole chiave e siti web

- Algoritmo: <https://it.wikipedia.org/wiki/Algoritmo>
- Bubblesort: [https://it.wikipedia.org/wiki/Bubble\\_sort](https://it.wikipedia.org/wiki/Bubble_sort)
- Timsort: <https://it.wikipedia.org/wiki/Timsort>
- O grande: <https://it.wikipedia.org/wiki/O-grande>





## 4. Serata cinematografica








Alcuni amici vogliono vedere un film insieme. È possibile scegliere tra sette film. Per prendere una decisione, ogni persona valuta ogni film come bello 😊, così così 😐 o brutto 😞.

I risultati sono visibili qui sotto. Purtroppo non c'è un vincitore, o un film «preferito», per la serata cinematografica.

Un film è un «preferito» se ogni persona ha dato a quel film la sua valutazione migliore. Ad esempio, il film 1 non è il preferito perché Niklaus ha dato il suo voto migliore a un altro film, il film 4.

Ora Ada vuole convincere il minor numero possibile di amici a cambiare la propria valutazione, in modo che alla fine ci sia un preferito.

*Aiuta Ada e modifica il minor numero possibile di valutazioni in modo che ci sia un preferito.*








	 1	 2	 3	 4	 5	 6	 7
Ada	😊	😊	😊	😊	😊	😊	😊
Nancy	😐	😊	😊	😐	😐	😊	😊
Niklaus	😞	😞	😞	😐	😞	😞	😞
Grace	😞	😐	😐	😐	😞	😐	😞
Edsger	😊	😐	😞	😞	😐	😊	😊
Rozsa	😐	😞	😐	😞	😊	😐	😐



## Soluzione

All'inizio non c'è un film preferito. Per ogni film troviamo amici che valutano meglio altri film.

### Film      Amici che valutano meglio altri film

 <b>1</b>	4: Nancy, Niklaus, Grace e Rozsa
 <b>2</b>	3: Niklaus, Edsger e Rozsa
 <b>3</b>	3: Niklaus, Edsger e Rozsa
 <b>4</b>	3: Nancy, Edsger e Rozsa
 <b>5</b>	3: Nancy, Grace e Edsger
 <b>6</b>	2: Niklaus e Rozsa
 <b>7</b>	3: Niklaus, Grace e Rozsa

Per il film 6, ci sono solo due amici che valutano meglio altri film. Per tutti gli altri film, ce ne sono più di due. Se un solo amico modifica una valutazione, non è possibile creare un preferito. Ada deve quindi convincere Niklaus e Rozsa a migliorare il loro voto per il film 6. Pertanto, Ada ha creato un preferito con due modifiche.

Il miglioramento delle valutazioni è una delle possibili strategie. Niklaus e Rozsa potrebbero ancora decidere di abbassare alcune valutazioni: se Niklaus peggiora la sua valutazione per il film 4 e Rozsa peggiora la sua valutazione per il film 5, il film 6 diventa il preferito. Anche in questo caso sono necessarie due modifiche.

È abbastanza plausibile che Rozsa peggiori la sua valutazione per il film 5 e che Niklaus migliori la sua valutazione per il film 6. Allo stesso modo, Rozsa potrebbe migliorare la sua valutazione per il film 6 e Niklaus potrebbe peggiorare la sua valutazione per il film 4. In entrambi gli scenari, il film 6 diventa il preferito. In entrambi i casi, sono sufficienti due modifiche.

In totale, quindi, ci sono quattro modi per modificare solo due valutazioni, in modo da avere un preferito.

## Questa è l'informatica!

Come possiamo risolvere questo compito? Un'idea è quella di verificare per ogni film e persona singolarmente se quella persona ha valutato altri film meglio o peggio. Nel nostro caso, si ottiene la tabella qui sopra. Questa tabella ci aiuta a capire quali persone devono modificare le loro valutazioni, in modo da arrivare a un preferito con il minor numero possibile di modifiche.

Ada può effettivamente usare questo *algoritmo* per risolvere il suo problema.



Tuttavia, questo algoritmo è *efficiente*? Ada potrebbe essere ancora più veloce?

Di seguito indichiamo il numero di film con  $M$  e il numero di amici con  $F$ . Ada deve considerare singolarmente tutte le  $M \times F$  valutazioni e per ogni valutazione deve considerare tutte le altre  $M - 1$  valutazioni della stessa persona. In totale, Ada deve considerare  $M \times (M - 1) \times F$  valutazioni.

Per scoprire se una delle valutazioni è problematica, Ada deve solo confrontare quella valutazione con la migliore che quella persona ha dato. Se quella persona pensa che un altro film sia migliore, allora il film che Ada ha appena guardato potrebbe non essere affatto il suo preferito.

In altre parole, se Ada scopre prima le migliori valutazioni complessive per ogni persona (esaminando tutte le  $M \times F$  valutazioni), può determinare per tutte le  $M \times F$  valutazioni se sono peggiori della migliore valutazione di quella persona.

Nel complesso, questo algoritmo alternativo con un pre-calcolo mirato delle migliori valutazioni porta Ada a considerare le valutazioni di  $2 \times M \times F$ . Con  $M = 7$  e  $F = 6$ , si tratta di 84 accessi alla tabella, mentre il primo algoritmo richiede 252 accessi alla tabella. Anche il secondo algoritmo risolve correttamente il problema di Ada, ma è più efficiente del primo.

Uno dei compiti più importanti dell'informatica è quello di risolvere i problemi non solo in modo corretto, ma anche nel modo più efficiente possibile. Con i computer più veloci le soluzioni vengono calcolate più rapidamente. Tuttavia, se non si conoscono algoritmi efficienti per risolvere un problema, anche i computer più veloci possono raggiungere i loro limiti.

## Parole chiave e siti web

- Algoritmo: <https://it.wikipedia.org/wiki/Algoritmo>



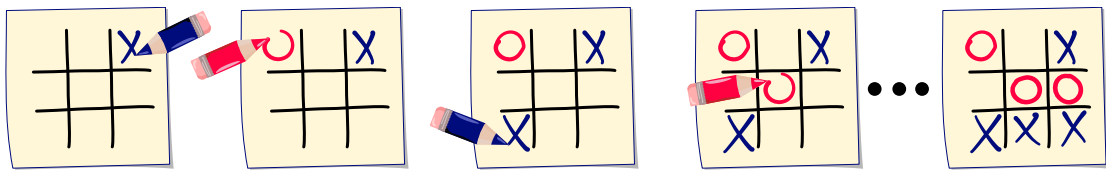


## 5. Tris

Il tris è un gioco per due persone.

In una griglia con  $3 \times 3$  caselle, i due giocatori riempiono a turno un simbolo in una casella vuota: un giocatore  $X$ , l'altro  $O$ . Il primo giocatore che riempie tre caselle in fila, in colonna o in diagonale con il proprio simbolo vince e la partita è finita. Se tutte le caselle sono riempite e nessuno ha vinto, la partita termina con un pareggio.

Qui si possono vedere i punteggi di una possibile partita: le prime 4 mosse e l'ultima mossa. Il giocatore con  $X$  vince.



Il punteggio alla fine di una partita è chiamato punteggio finale. Le regole del gioco specificano esattamente come possono essere compilati i campi con  $X$  e  $O$  e quando il gioco termina.

*Solo una delle quattro immagini mostra un punteggio finale di tris. Quale?*

- A) 

X	O	X
O	X	O
O	O	X
- B) 

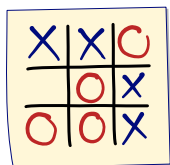
X	O	X
O	X	
O	X	X
- C) 

X	X	O
	O	X
O	O	X
- D) 

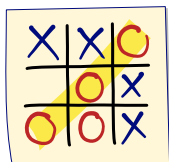
X	O	X
O	X	O
O	X	



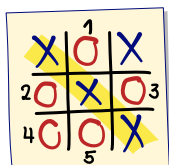
## Soluzione



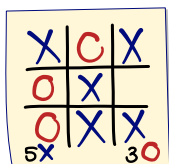
La risposta C è corretta:



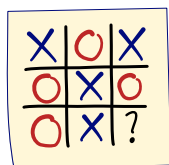
La risposta C è corretta perché un giocatore aveva vinto (tre in una diagonale) e poi non sono state fatte altre mosse.



La risposta A non è corretta. Il giocatore ha vinto la partita, ma il giocatore ha riempito più caselle. Poiché il vincitore riempie sempre l'ultimo campo, non potrà mai avere meno caratteri del perdente.



La risposta B non è corretta perché 5 campi sono riempiti con ma solo 3 campi con . Questo non è possibile, perché il numero di caratteri e il numero di caratteri possono differire al massimo di 1.



La risposta D non è corretta, perché non mostra un punteggio finale. Non c'è ancora un vincitore e i campi non sono completamente riempiti.

## Questa è l'informatica!

Nel risolvere il compito, abbiamo verificato se le quattro immagini delle opzioni di risposta documentano una posizione finale valida. Dalle regole del gioco del tris si possono ricavare nuove regole sulle posizioni finali valide, ad esempio queste:

1. La differenza tra il numero di e il numero di deve essere pari a 0, -1 o 1.
2. Se nessun giocatore ha vinto, tutte le caselle devono essere riempite.
3. Il perdente può compilare al massimo tanti campi quanti ne ha compilati il vincitore.
4. Nel documento di un gioco finito, può esserci al massimo una sequenza di tre caratteri uguali.

Queste nuove regole non sono regole del gioco, ma servono solo a verificare se la griglia completata è un punteggio finale. Se un'immagine è in conflitto con una di queste regole, non può costituire un punteggio finale.

Le regole sono molto importanti nella tecnologia informatica. Un interprete che esegue un programma controlla se il testo inserito è conforme alle regole di sintassi del linguaggio di programmazione.

Nella programmazione, le regole vengono utilizzate nelle cosiddette assicurazioni per verificare la correttezza di un programma durante la sua esecuzione.



## Parole chiave e siti web

- Tris: [https://it.wikipedia.org/wiki/Tris\\_\(gioco\)](https://it.wikipedia.org/wiki/Tris_(gioco))
- Interprete: [https://it.wikipedia.org/wiki/Interprete\\_\(informatica\)](https://it.wikipedia.org/wiki/Interprete_(informatica))
- Linguaggio di programmazione:  
[https://it.wikipedia.org/wiki/Linguaggio\\_di\\_programmazione](https://it.wikipedia.org/wiki/Linguaggio_di_programmazione)

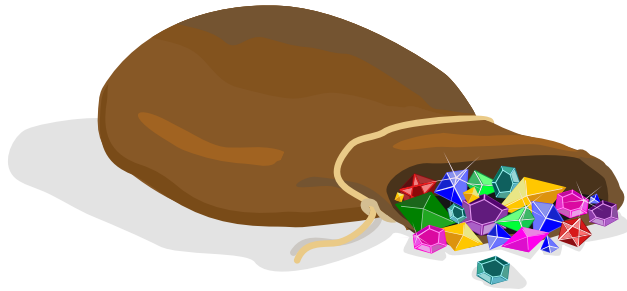




## 6. Pietre preziose

Pietro ha delle pietre preziose.  
Hanno tutte un valore diverso.

Sarah conosce le pietre preziose  
di Pietro, ma non il loro valore.  
Vuole sapere qual è la pietra più  
preziosa.



A tal fine, esegue tre volte la  
seguinte procedura:

- Sceglie quattro pietre di Pietro e gli chiede quale sia la più preziosa.

Ogni volta sceglie le quattro pietre a caso e Pietro le dà ogni volta una risposta sincera.

Dopodiché, Sarah sa qual è la pietra più preziosa.

*Quante pietre preziose può avere al massimo Pietro?*

- A) 8 pietre preziose
- B) 10 pietre preziose
- C) 11 pietre preziose
- D) 12 pietre preziose



## Soluzione

La risposta B) è corretta: 10 pietre preziose

Se Pietro ha 10 pietre, Sarah può scegliere un totale di otto pietre diverse nelle prime due domande. Le due «vincitrici» delle singole domande (cioè le pietre più preziose tra le quattro scelte) possono anche essere «vincitrici complessive», cioè la pietra di maggior valore in assoluto. Le altre sei pietre vengono eliminate. Per l'ultima domanda, sceglie i due vincitori e le due pietre non ancora scelte. Il vincitore di questa domanda deve essere la pietra con il maggior valore.

Quindi, per 10 pietre, Sarah può (tra le altre cose) procedere in questo modo per trovare la pietra più preziosa. Se Pietro ha 11 pietre, purtroppo non può farlo.

Se, come sopra, Sarah confronta un totale di otto pietre diverse nelle prime due domande, le rimangono le due pietre con il valore più alto e altre tre pietre, una di troppo, per trovare il vincitore assoluto con la terza domanda. Se, invece, Sarah confronta il vincitore della prima domanda con le 3 «nuove» pietre della seconda domanda, allora conosce la più preziosa delle sette pietre scelte. Deve confrontare questa pietra con le altre quattro. Anche questa è una pietra di troppo per la terza domanda.

Se Sarah sceglie solo sei o anche meno pietre diverse per le prime due domande con 11 pietre, o se Pietro ha più di 12 pietre, Sarah non può sapere quale pietra è la più preziosa dopo tre domande.

## Questa è l'informatica!

Questo compito riguarda un *algoritmo* vincolato da condizioni. Nel nostro caso, Sarah può porre solo tre domande e ogni domanda può contenere solo 4 elementi.

Nonostante questa restrizione, l'algoritmo funziona bene per raccolte di dimensioni inferiori a 11, ma non funziona altrimenti.

Le ragioni per imporre restrizioni agli algoritmi possono essere varie. Ad esempio, si potrebbe richiedere che un'operazione venga completata in un tempo fisso, come avviene nei sistemi operativi in tempo reale. Un altro motivo potrebbe essere che le operazioni possono comportare costi esterni o danneggiare un componente.

Non è un problema se l'algoritmo fallisce al di sopra di una certa soglia, purché sia garantito che tale soglia non venga mai raggiunta. Ad esempio, la strategia ristretta di questo compito non deve mai essere utilizzata per collezioni superiori a 10.

## Parole chiave e siti web

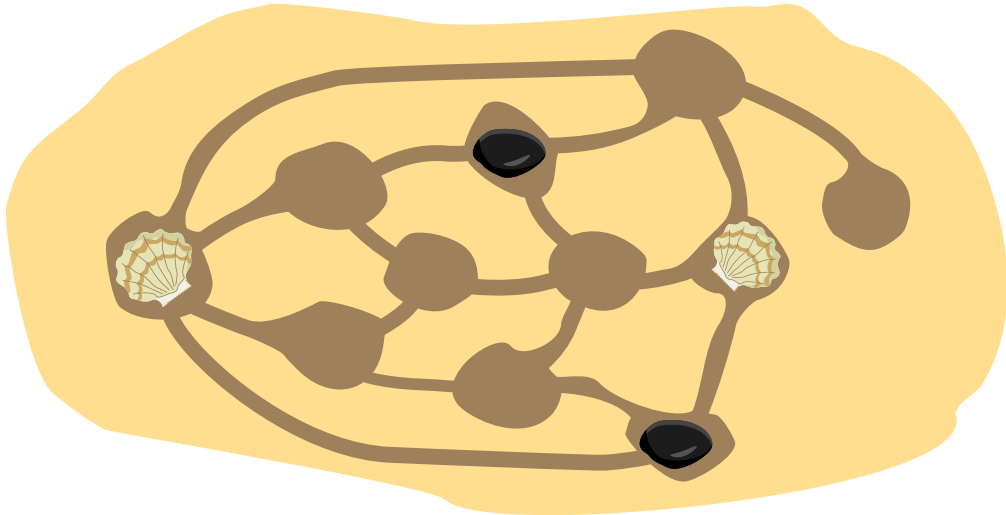
- Algoritmo: <https://it.wikipedia.org/wiki/Algoritmo>
- Complessità temporale: [https://it.wikipedia.org/wiki/Complessità\\_temporale](https://it.wikipedia.org/wiki/Complessità_temporale)



## 7. Ciottoli e conchiglie

Ann e Bob giocano sulla spiaggia. Scavano delle cavità e ne collegano alcune con solchi disegnati sulla sabbia. Le pedine di Ann sono conchiglie 🐚. Quelle di Bob sono ciottoli ⬛.

A turno, i giocatori posizionano uno delle loro pedine in uno spazio libero. Il primo giocatore che posiziona due proprie pedine in due cavità direttamente collegate perde. L'immagine mostra il punteggio dopo alcune mosse.

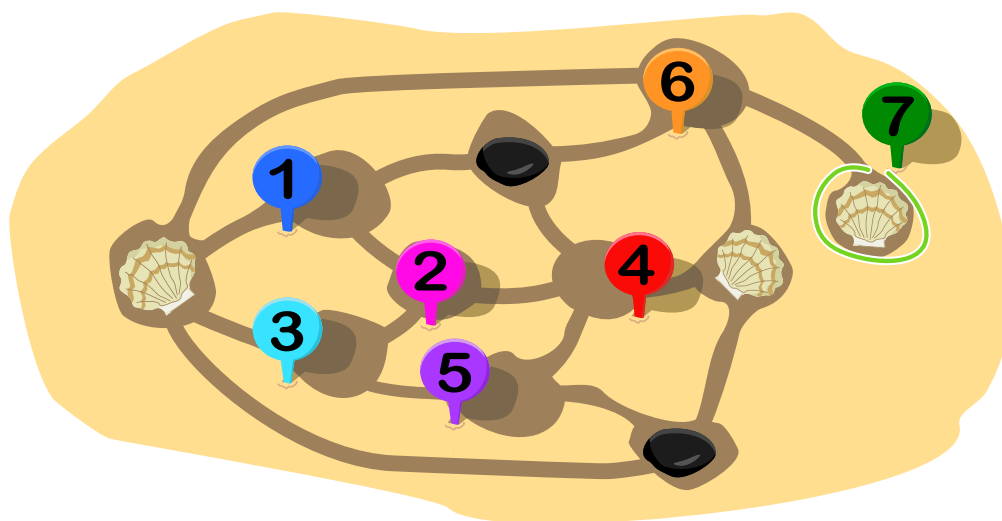


*È il turno di Ann. In quale delle cavità libere deve posizionare la sua prossima conchiglia per assicurarsi la vittoria?*

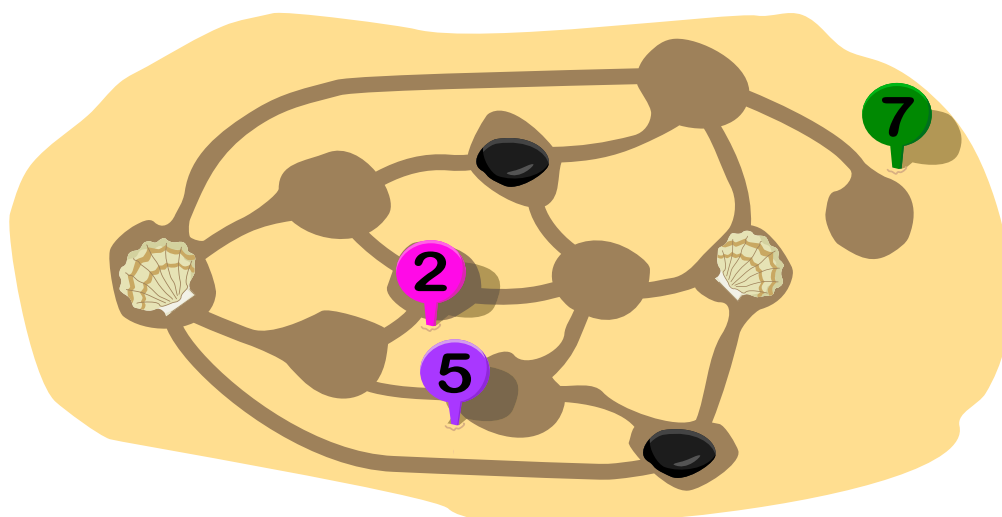


## Soluzione

La risposta corretta è la cavità 7.



È il turno di Ann. Per lei, le cavità 1, 3, 4 e 6 sono fuori discussione, quindi restano la 2, la 5 e la 7.



Vede che per Bob le cavità 1, 4, 5 e 6 sono fuori discussione. Quindi per lui rimangono 2, 3 e 7.

Se Ann gioca 7, Bob può giocare 2 o 3; in entrambi i casi Ann può comunque giocare 5 e Bob perde.

Se Ann giocasse 2 al punteggio della figura, Bob potrebbe giocare 7 al prossimo colpo. Dopo di che, Ann avrebbe dovuto giocare 5, Bob avrebbe dovuto giocare 3 e Ann avrebbe perso.

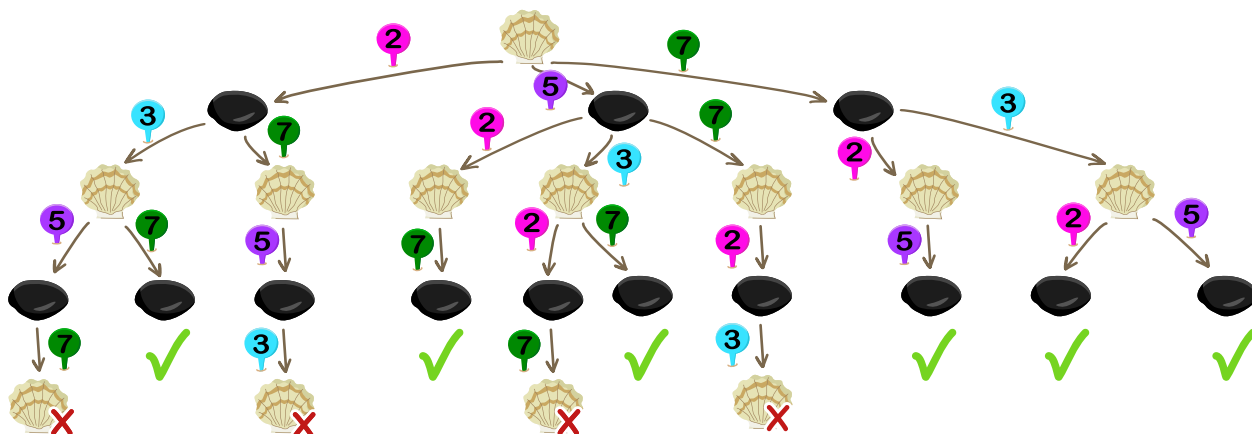
Se Ann giocasse 5, Bob potrebbe giocare 7, Ann dovrebbe giocare 2, Bob giocherebbe 3 e di nuovo Ann perderebbe.

Tra l'altro, Bob non potrebbe vincere nemmeno se fosse il suo turno al punteggio nella foto.



## Questa è l'informatica!

Per visualizzare sistematicamente le possibili mosse di Ann e Bob, si può utilizzare un cosiddetto albero di gioco:



In questo albero di gioco si può vedere con quale mossa Ann può assicurarsi la vittoria: nel ramo di destra, che inizia con Ann che gioca 7, sono possibili solo situazioni in cui vince. Nella cosiddetta *teoria dei giochi*, un campo speciale della matematica, si considerano le affermazioni sull'esito dei giochi in cui interagiscono due o più giocatori. L'informatica si occupa di algoritmi per la valutazione di tali alberi di gioco. I computer con una potenza di calcolo sufficiente possono già competere con gli esseri umani in giochi come gli scacchi e vincere. Tuttavia, la teoria dei giochi fornisce anche alla psicologia, all'economia e ad altre materie modelli per sistemi complessi in cui i «giocatori» interagiscono, ad esempio per il comportamento di acquisto dei clienti quando i prezzi cambiano o per la selezione del percorso nel traffico stradale.

Il gioco di Ann e Bob è un'istanza di «COL». Si tratta di un gioco per due giocatori introdotto da Colin Vout e descritto nel noto libro «On Numbers and Games» del matematico John Horton Conway.

## Parole chiave e siti web

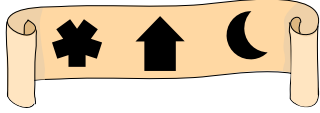
- Teoria dei giochi: [https://it.wikipedia.org/wiki/Teoria\\_dei\\_giochi](https://it.wikipedia.org/wiki/Teoria_dei_giochi)
- John Horton Conway: [https://it.wikipedia.org/wiki/John\\_Conway](https://it.wikipedia.org/wiki/John_Conway)



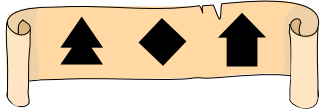


## 8. Maria alla caccia del tesoro

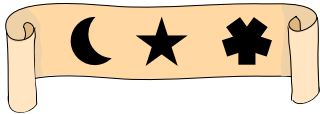
Maria trova una scatola misteriosa. Purtroppo la scatola è chiusa a chiave. Per aprirla, Maria deve scoprire la «chiave»: la giusta combinazione di tre simboli. Per fortuna, accanto alla scatola trova anche gli indizi di alcune combinazioni sbagliate:



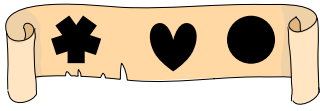
1) Uno dei simboli fa parte della chiave e si trova nella posizione giusta.



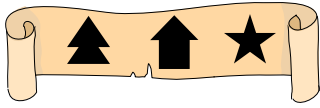
2) Nessuno dei simboli fa parte della chiave.



3) Due simboli fanno parte della chiave. Ma entrambi sono nella posizione sbagliata.



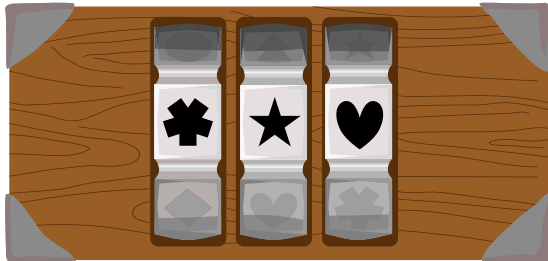
4) Un simbolo fa parte della chiave. Ma questo simbolo è nella posizione sbagliata.



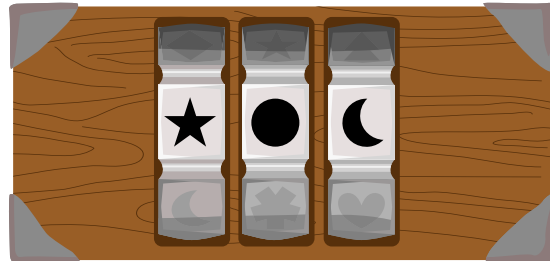
5) Un simbolo fa parte della chiave. Ma è nella posizione sbagliata.

Una delle seguenti combinazioni è la chiave della scatola. Quale?

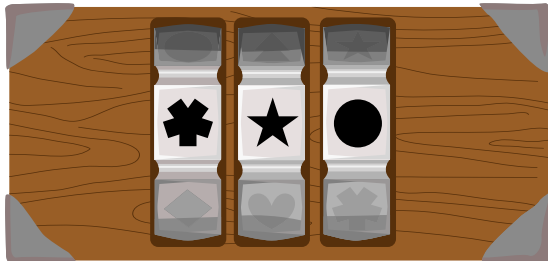
A)



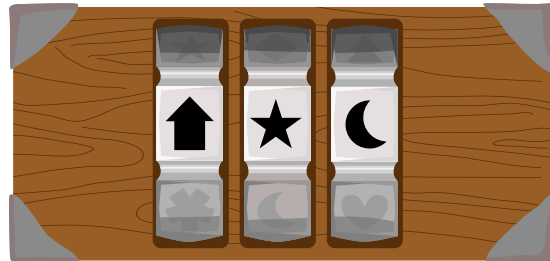
B)



C)










D)





## Soluzione



La risposta corretta è B).

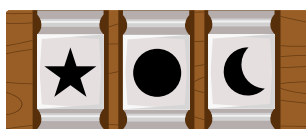
Cominciamo con l'individuare i simboli che possono essere presenti nella chiave. Dopo l'indizio 2) possiamo eliminare i simboli che non possono far parte della chiave: l'abete , il diamante  e la casa . Il suggerimento 5) indica che un simbolo fa parte della chiave ma si trova nella posizione sbagliata. Poiché l'abete  e la casa  non possono essere presenti nella chiave, sappiamo che la stella  fa parte della chiave ma è nella posizione sbagliata. La nota 3) esclude che la stella  possa trovarsi al centro. In questo modo conosciamo la posizione finale della stella:



Poiché esiste una sola risposta possibile, che inizia con la stella, abbiamo già trovato la chiave. Per convincerci, continuiamo a cercare i due simboli mancanti. L'indizio 1) mostra che un simbolo compare nella chiave ed è già nella posizione giusta. La casa `house` e la prima posizione potrebbero già essere escluse. Pertanto, sappiamo che la luna si trova nella posizione corretta. Il risultato è la seguente immagine:



La nota 4) indica che un simbolo fa parte della chiave ma si trova nella posizione sbagliata. Possiamo escludere il simbolo . Inoltre, solo lo spazio centrale è ancora libero. Pertanto, anche il cuore  non può far parte della chiave. Ne consegue che il cerchio assume la posizione centrale.



Il risultato corretto può essere determinato anche in un altro modo. Tuttavia, ogni possibilità porta allo stesso risultato.

## Questa è l'informatica!

Questo compito può essere risolto logicamente, ad esempio con l'aiuto del «metodo dell'esclusione». Nel nostro caso, siamo partiti dall'indizio 2) e abbiamo escluso tre simboli, che ci hanno portato rapidamente alla chiave. Le priorità dell'indizio 2) potrebbero essere viste come strategie mentali, regole o scorciatoie che ci hanno aiutato a prendere una decisione con conoscenze e tempo limitati. In informatica, tali regole sono chiamate *euristiche*, che possono anche essere programmate e automatizzate.



Ogni giorno prendiamo molte piccole decisioni basate su indizi o sulla necessità di comprendere vari *vincoli* di un problema per risolverlo. In questo compito, abbiamo seguito gli indizi e risolto il problema passo dopo passo per aprire la scatola.

Come potrebbe un computer risolvere questo problema? Questi otto simboli possono essere disposti in tre posizioni in un totale di 336 modi. Un computer le proverebbe tutte. In informatica si parla di *ricerca completa*. Una ricerca completa (chiamata anche *brute force* o *recursive backtracking*) è un metodo di risoluzione di un problema in cui viene attraversato l'intero spazio di ricerca. A noi questa soluzione può sembrare molto *inefficiente*, perché avremmo bisogno di molto tempo per provare tutte le possibilità (e dimenticare quelle già provate). Tuttavia, un computer può risolvere tali compiti in modo molto rapido e quindi efficiente. I simboli dell'esempio potrebbero anche rappresentare una password. Inoltre, la password deve essere sempre scelta in modo che contenga il maggior numero possibile di caratteri diversi, in modo che anche una ricerca completa non produca la chiave in un tempo ragionevole.

Iniziare con il suggerimento 2), quindi minimizzare le possibili soluzioni, è chiamato in informatica *backtracking*. Ad ogni *vertice* di un *albero*, vengono eliminate le possibilità che ovviamente non possono verificarsi nella chiave. In questo modo, a ogni profondità dell'albero, le possibilità si riducono.

## Parole chiave e siti web

- Euristiche: <https://it.wikipedia.org/wiki/Euristica>
- Vertice: [https://it.wikipedia.org/wiki/Vertice\\_\(teoria\\_dei\\_grafi\)](https://it.wikipedia.org/wiki/Vertice_(teoria_dei_grafi))
- Albero: [https://it.wikipedia.org/wiki/Albero\\_\(grafo\)](https://it.wikipedia.org/wiki/Albero_(grafo))



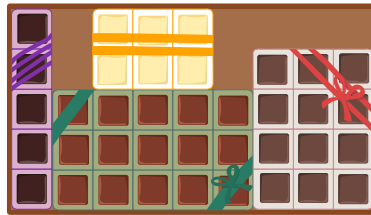


## 9. Incarto di cioccolatini

La fabbrica di cioccolato «Castocolat» invia quattro scatole di cioccolatini a ciascuno dei suoi clienti per una campagna pubblicitaria.

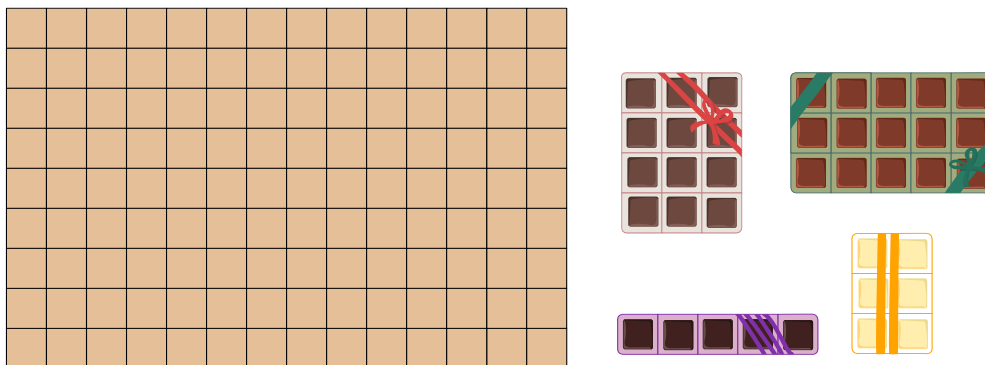
Per risparmiare spese postali e materiale, Linus deve mettere le quattro scatole diverse una accanto all'altra in una cassetta più piccola possibile. Le scatole non devono essere impilate l'una sull'altra, altrimenti le praline si schiacciano.

Linus ha disposto le praline in questo modo, in una cassetta per  $5 \times 9 = 45$  praline individuali.



Lina però dice a Linus: «Se metti le scatole in modo diverso, entreranno in una cassetta più piccola.»

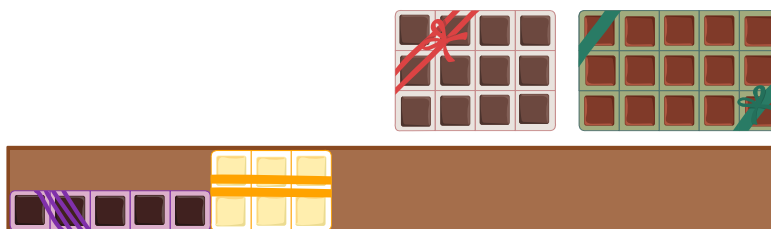
*Posiziona le scatole in modo che entrino in una cassetta più piccola possibile.*





## Soluzione

In totale ci sono  $12 + 15 + 6 + 5 = 38$  cioccolatini che Linus deve mettere in una cassetta. Una cassetta in cui si possono inserire 38 praline singole senza alcuno spazio vuoto deve avere una dimensione di  $1 \times 38$  o  $2 \times 19$  (2 e 19 sono gli unici divisori di 38). Le due scatole di praline di dimensioni  $3 \times 4$  e  $3 \times 5$  non entrerebbero in nessuna di queste cassette.



Se Linus sceglie una cassetta per 39 praline (cioè con uno spazio vuoto per esattamente un'altra pralina), essa ha le dimensioni  $1 \times 39$  o  $3 \times 13$ . Le cassette  $3 \times 5$ ,  $3 \times 4$ ,  $3 \times 2$  entrano nella cassetta, ma la scatola  $1 \times 5$  non entra nello spazio libero rimanente di dimensione  $2 \times 3$ .



Una cassetta per 40 cioccolatini può avere le seguenti dimensioni  $1 \times 40$ ,  $2 \times 20$ ,  $4 \times 10$ ,  $5 \times 8$ . Non tutte le scatole entrano nelle cassette con le dimensioni  $1 \times 40$  o  $2 \times 20$ . Nelle altre due caselle si inseriscono tutte e quattro le caselle, ad esempio in questo modo:



Puoi aggiungere le scatole ad altre composizioni che entrano in una cassetta per 40 praline. Quindi queste quattro scatole di praline non possono essere confezionate in modo più salvaspazio che con lo spazio vuoto per 2 cioccolatini.

## Questa è l'informatica!

In questo compito, i rettangoli devono essere disposti in modo tale che il rettangolo che li racchiude abbia l'area minima. Questo problema è noto in informatica anche come «impacchettamento dei rettangoli», uno dei tanti cosiddetti problemi di impacchettamento. Per alcuni rettangoli possiamo trovare la soluzione *ottimale* in modo relativamente semplice (in questo caso la cassetta più piccola possibile). Per quantità maggiori, è necessario automatizzare il processo; quindi è necessario un algoritmo che possa essere realizzato come programma informatico. Sfortunatamente, l'«impacchettamento dei rettangoli», come molti altri problemi di impacchettamento, è *NP-completo*. Questo significa che molto probabilmente non esiste un *algoritmo efficiente* per il problema che trovi



soluzioni ottimali. In informatica vengono quindi sviluppati algoritmi efficienti per i problemi NP-completi, che non garantiscono di trovare soluzioni ottimali, ma possono trovarne di sufficientemente buone.

Per le aziende di logistica, tra le altre cose, le soluzioni efficienti per problemi di questo tipo sono di grande importanza, ad esempio per lo stoccaggio in scaffali alti, per l'imballaggio salvaspazio delle merci o per la distribuzione delle merci nei container. Inoltre, problemi apparentemente diversi possono essere descritti come problemi di impacchettamento. Ad esempio, un processo di lavoro che  $N$  lavoratori possono gestire in  $M$  ore può essere rappresentato come un rettangolo  $N \times M$ . In questo modo è possibile gestire diversi processi con il minor dispendio di persone e di tempo possibile se il problema dell' «impacchettamento dei rettangoli» viene risolto in modo ottimale per i rettangoli corrispondenti.

## Parole chiave e siti web

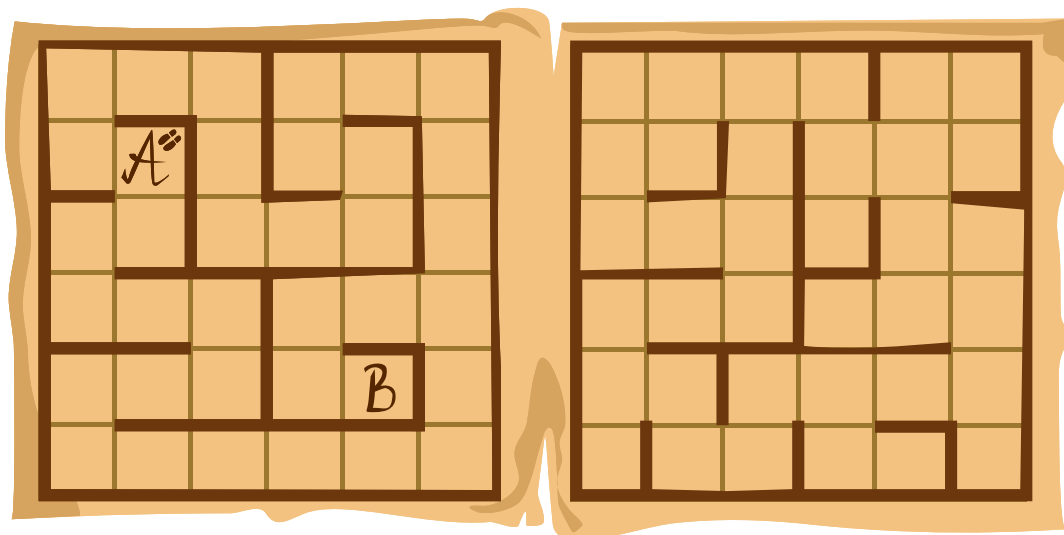
- NP-completo: <https://it.wikipedia.org/wiki/NP-completo>
- Ottimizzazione : [https://it.wikipedia.org/wiki/Ottimizzazione\\_\(matematica\)](https://it.wikipedia.org/wiki/Ottimizzazione_(matematica))





## 10. Labirinto

La scuola di magia ha due piani. I piani sono esattamente uno sopra l'altro. Entrambi sono divisi in campi e tra alcuni di essi ci sono dei muri:



Lo studente mago Ron ha bisogno di 1 secondo per passare da una casella all'altra sullo stesso piano. Purtroppo Ron ha dimenticato come attraversare i muri. Tuttavia, può passare da un piano al quadrato corrispondente dell'altro piano, impiegando 5 secondi.

Ron vuole passare dalla casella A alla casella B il più velocemente possibile.

*Di quanti secondi ha bisogno Ron al minimo per farlo?*

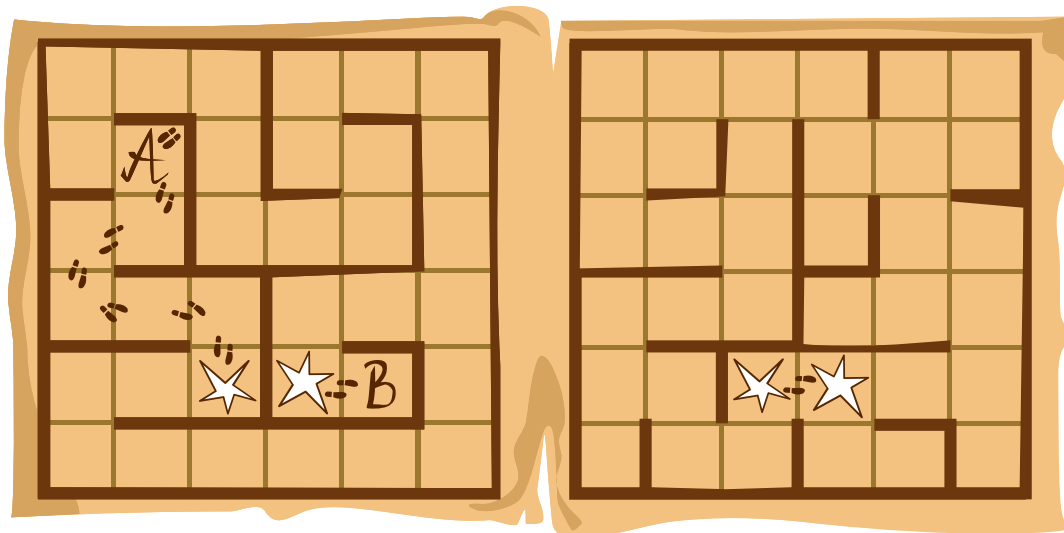
- A) 6 secondi
- B) 16 secondi
- C) 18 secondi
- D) 20 secondi



## Soluzione

La risposta C) 18 secondi è corretta.

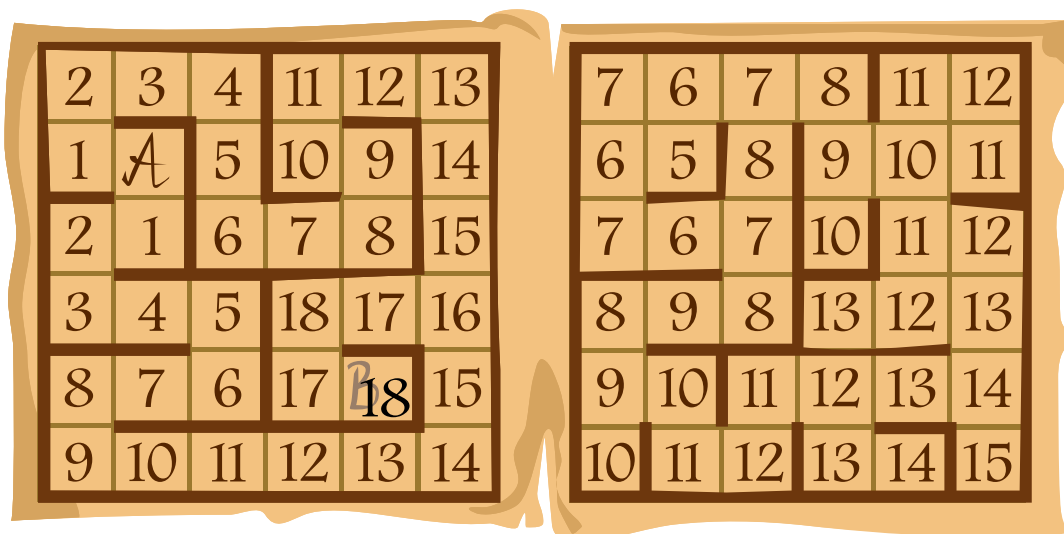
Quindi Ron può andare da A a B in 18 secondi:



È questo il modo più veloce? Il «tempo più breve» che Ron impiega per andare dal campo A a qualsiasi altro campo può essere calcolato passo dopo passo in questo modo:

Per il campo A, il tempo più breve è ovviamente 0 secondi. Poi continua passo dopo passo in questo modo: tra tutti i campi per i quali è già stato inserito il tempo più breve, scegli quello con il valore più basso. A partire da questo campo scelto, si esaminano tutti i possibili campi successivi e si considera come arrivarci più velocemente dal campo scelto; si inseriscono i tempi calcolati nei campi successivi. Può accadere che un tempo precedentemente inserito venga migliorato. In seguito, il campo selezionato non potrà più essere preso in considerazione e quindi non potrà più essere selezionato nelle fasi successive.

Ecco i tempi più brevi calcolati con questo metodo, partendo dal campo A:





Quindi Ron ha bisogno di almeno 18 secondi per andare dal campo A al campo B. 6 secondi (risposta A) sarebbe la durata del percorso più breve se non ci fossero muri tra i campi. Se poi Ron passasse da un piano all'altro, ci vorrebbero 16 secondi (risposta B). Se ci fosse solo il piano con i campi A e B, 20 secondi (risposta D) sarebbe il tempo più breve per il percorso da A a B.

## Questa è l'informatica!

I percorsi più veloci o più brevi devono essere calcolati molto spesso; un esempio ovvio è la pianificazione del percorso nelle moderne app di mappe. Il problema si semplifica notevolmente se i percorsi sono costituiti da singoli passaggi tra punti vicini e se per tutti questi passaggi è noto il loro «costo»: Tempo, denaro, consumo di energia - qualunque sia la quantità importante per il problema attuale. In questo caso, i punti, i passi e i costi dei passi possono essere astratti in un *grafo* in cui i passi possono essere messi insieme per formare dei percorsi. Per i grafi, sono noti molti algoritmi in informatica con cui è possibile calcolare in modo efficiente i *cammini minimi*. Uno di questi è stato inventato dall'informatico Edsger Dijkstra; questo *algoritmo di Dijkstra* è stato utilizzato sopra nella spiegazione della risposta corretta.

I percorsi più brevi giocano un ruolo importante anche nella progettazione di circuiti per computer. I punti di commutazione devono essere cablati insieme al minor costo possibile. I circuiti moderni sono costituiti da più livelli e il cablaggio tra due livelli è più costoso rispetto al cablaggio (altrimenti comparabile) sullo stesso livello - simile al passaggio da un piano all'altro in questo compito, che è più costoso di un passo sullo stesso piano.

## Parole chiave e siti web

- Grafo: <https://it.wikipedia.org/wiki/Grafo>
- Cammino minimo: [https://it.wikipedia.org/wiki/Cammino\\_minimo](https://it.wikipedia.org/wiki/Cammino_minimo)
- Edsger Dijkstra: [https://it.wikipedia.org/wiki/Edsger\\_Dijkstra](https://it.wikipedia.org/wiki/Edsger_Dijkstra)
- Algoritmo di Dijkstra: [https://it.wikipedia.org/wiki/Algoritmo\\_di\\_Dijkstra](https://it.wikipedia.org/wiki/Algoritmo_di_Dijkstra)



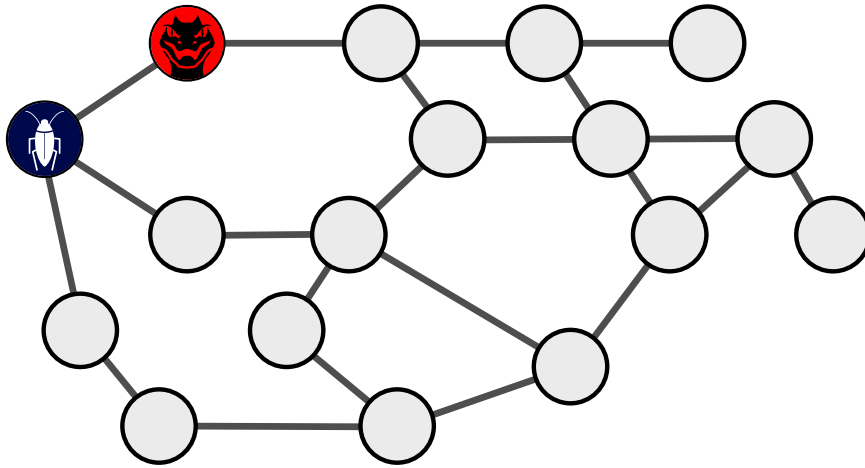


# 11. Virus

In una rete di computer, due nodi della rete sono stati infettati da virus informatici: uno con il virus BlueBug 🐛, l'altro con il virus RedRaptor 🦇. Entrambi i virus si diffondono sempre al mattino. Ogni virus infetta poi tutti i nodi che sono direttamente collegati ai nodi che ha già infettato. Se un nodo è infettato da entrambi i virus, si spegne dopo qualche ora a causa del sovraccarico 🚫. I virus non possono quindi diffondersi ulteriormente nei giorni successivi.

Qui sotto puoi vedere la rete di computer con i nodi e le loro connessioni dirette. I due nodi infettati all'inizio sono contrassegnati. Dopo qualche giorno, tutti i nodi vengono infettati da un virus o addirittura spenti.

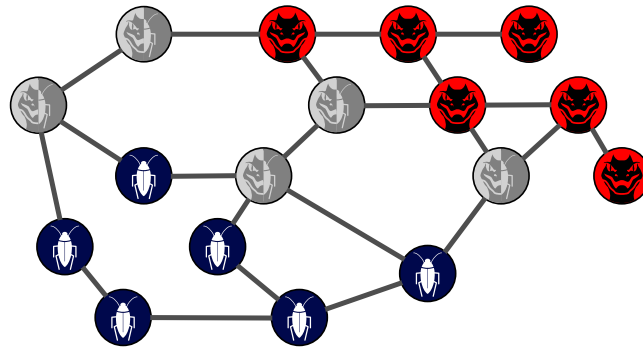
*Quali nodi vengono poi infettati da quale virus o spenti? Scegli il marcatore corretto per ogni nodo.*



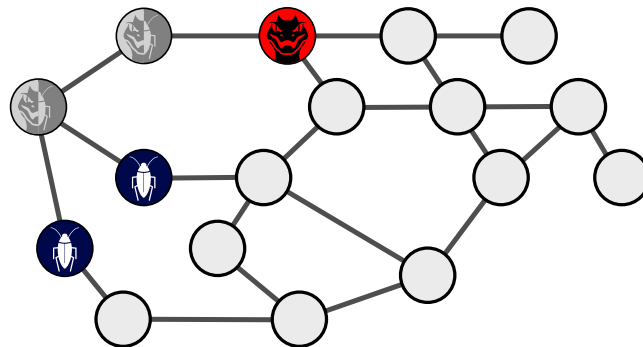


## Soluzione

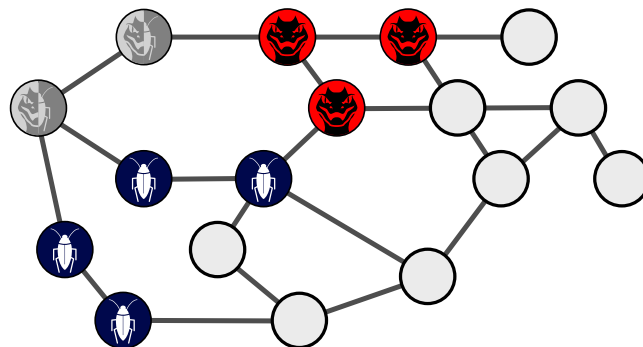
Dopo 5 giorni, tutti i nodi della rete vengono infettati o spenti. Questa è la soluzione corretta:



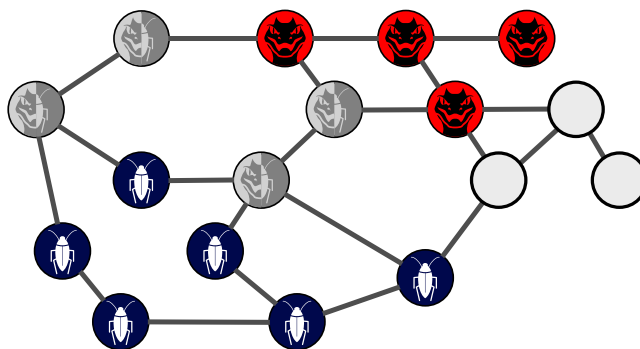
Dopo 1 giorno, cinque nodi della rete sono stati infettati. I due nodi infettati all'inizio sono ora infettati da entrambi i virus e quindi spenti:



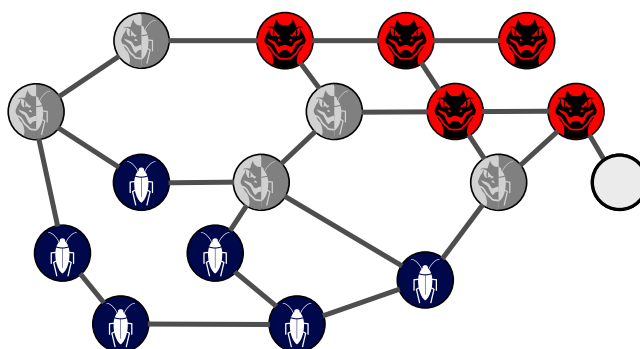
Dopo 2 giorni, vengono infettati altri quattro nodi:



Dopo 3 giorni, due nodi sono doppiamente infetti e ora sono anche spenti. Inoltre, altri tre nodi sono stati infettati da «BlueBug» e due da «RedRaptor»:



Dopo 4 giorni, un altro nodo della rete viene spento. «BlueBug» non può più diffondersi ulteriormente.



Il 5° giorno, l'ultimo nodo viene infettato da «RedRaptor».

## Questa è l'informatica!

I virus e le altre minacce informatiche rappresentano una grande sfida per le reti informatiche. Non solo influiscono sulle prestazioni dei computer colpiti, ma spesso hanno un «carico aggiuntivo» (*payload*) che causa ulteriori danni. In alcuni casi, ad esempio, i dati trasmessi vengono letti e quindi informazioni sensibili come password o dati utente vengono scoperti e trasmessi a un cliente. In alcuni casi, il virus cripta i dati presenti sul computer infetto. Se l'utente vuole accedere nuovamente ai suoi dati, deve prima trasferire una somma di denaro su un conto anonimo. A volte gruppi di computer infetti sono controllati a distanza da criminali per effettuare attacchi ad altri computer (*botnet*).

Il fatto che un virus paralizzi completamente un computer di solito non è voluto dal creatore del virus, perché questo blocca la diffusione del virus. Tuttavia, alcuni virus sono stati sviluppati appositamente per il sabotaggio e la guerra informatica. Questo può anche danneggiare in modo permanente i computer colpiti.

L'installazione degli ultimi aggiornamenti di sicurezza è un requisito importante per difendersi dai virus. I programmi antivirus possono migliorare la protezione, ma sono già inclusi in alcuni sistemi operativi, quindi un programma aggiuntivo potrebbe non essere necessario. Tuttavia, sono indispensabili backup regolari dei dati e un'attenta vigilanza sui comportamenti insoliti del sistema.



## Parole chiave e siti web

- Rete di computer: [https://it.wikipedia.org/wiki/Rete\\_di\\_computer](https://it.wikipedia.org/wiki/Rete_di_computer)
- Virus: [https://it.wikipedia.org/wiki/Virus\\_\(informatica\)](https://it.wikipedia.org/wiki/Virus_(informatica))
- Botnet: <https://it.wikipedia.org/wiki/Botnet>
- Guerra cibernetica: [https://it.wikipedia.org/wiki/Guerra\\_cibernetica](https://it.wikipedia.org/wiki/Guerra_cibernetica)

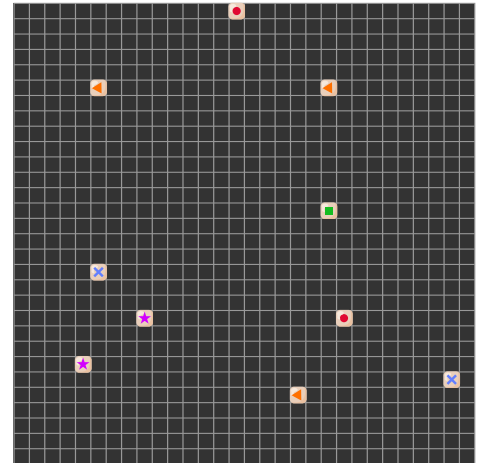


## 12. Colorazione del pavimento

Il pavimento di una stanza quadrata è diviso in  $30 \times 30$  caselle. Su dieci caselle si trovano i gettoni con tali simboli colorati: , , , e .

Un robot deve dipingere il pavimento con questi simboli, campo per campo. A tal fine, sono previste quattro regole diverse. Su un campo dove non ci sono gettoni, si dipinge ...

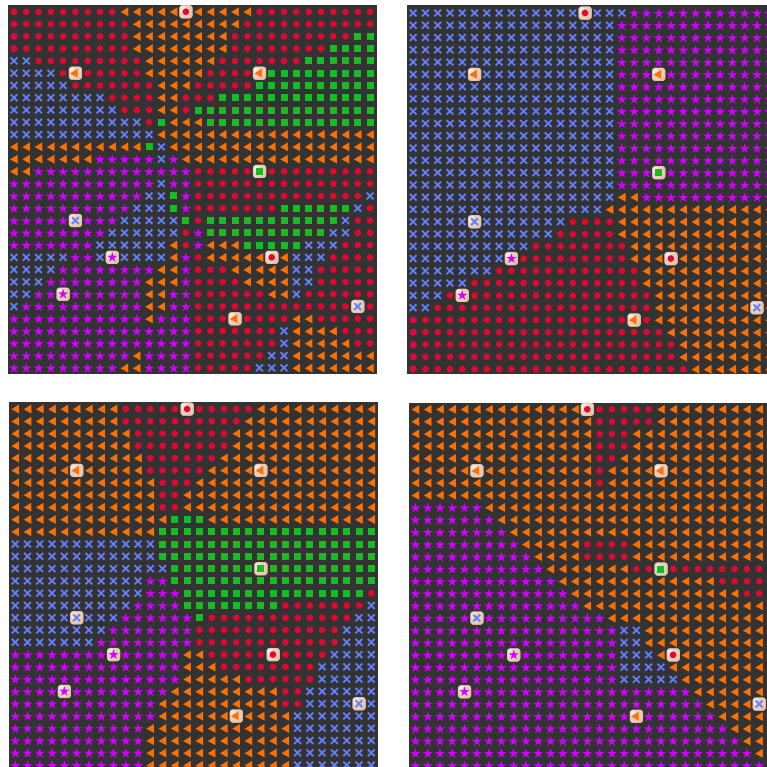
- 1 ... il simbolo del gettone più vicino a lui.
- 2 ... il simbolo del gettone più lontano da lui.
- 3 ... il simbolo del gettone che è il secondo più vicino a lui.
- 4 ... il simbolo che si ripete più frequentemente tra i 6 gettoni più vicini.



Il robot dipinge tutti i quadrati seguendo la stessa regola. Se la regola di un campo fornisce diversi simboli possibili, il robot ne sceglie uno a caso.

Di seguito è possibile vedere come viene dipinto il pavimento alla fine per ogni regola.

*Quale pavimento corrisponde a quale regola? Abbina le regole ai pavimenti.*

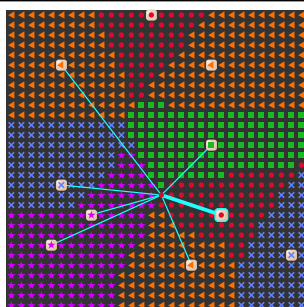




## Soluzione

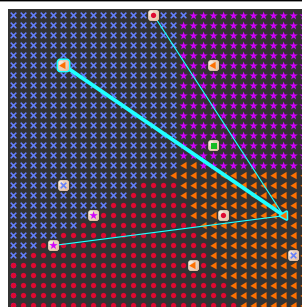
Poiché tutti i campi di un pavimento sono dipinti secondo la stessa regola, è sufficiente controllare un campo alla volta. Per ogni pavimento esaminiamo un campo diverso:

Regola 1



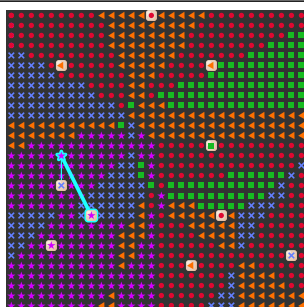
Il campo è dipinto con ● perché un gettone ● è il più vicino.

Regola 2



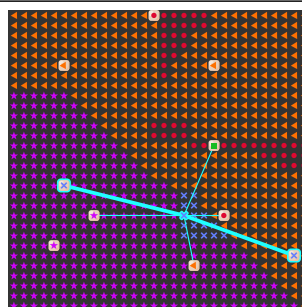
Il campo è dipinto con ◀ perché un gettone ◀ è il più lontano.

Regola 3



Il campo è dipinto con ★ perché un gettone ★ è il secondo più vicino.

Regola 4



Il campo viene dipinto con × perché questo × è il più comune tra i 6 gettoni più vicini.

## Questa è l'informatica!

Le divisioni di un piano e la loro costruzione *algoritmica* svolgono un ruolo importante in varie aree dell'informatica, ad esempio nelle simulazioni e nella computer grafica.

I *diagrammi di Voronoi*, che prendono il nome dal matematico ucraino Georgi Feodosyevich Voronoi (\*1868 - †1908), dividono un piano in regioni attorno ai cosiddetti *centri*. Tutti i punti di una regione non sono più vicini a nessun altro centro del proprio. Il risultato della regola 1 è un diagramma di Voronoi. Questi diagrammi si trovano spesso in situazioni reali, ad esempio nella copertura della telefonia mobile. Oppure vengono utilizzati nell'analisi delle partite di calcio o di altri comportamenti socio-economici, come le relazioni tra la popolazione e le scuole, gli ospedali o alcuni fornitori di servizi più vicini.

Nel 1911, il meteorologo Alfred H. Thiessen (\*1872 - †1956) ha sviluppato un metodo per determinare i valori medi (ad esempio le quantità di precipitazioni) delle aree in modo più realistico con l'aiuto



dei diagrammi di Voronoi. Non determina la media dei valori misurati dalle stazioni di misura solo in base al numero di stazioni di misura, ma utilizza il diagramma di Voronoi per determinare innanzitutto l'area a cui si riferiscono i valori misurati. Ciò comporta una diversa ponderazione dei valori misurati a livello locale.

## Parole chiave e siti web

- Algoritmo: <https://it.wikipedia.org/wiki/Algoritmo>
- Diagramma di Voronoi: [https://it.wikipedia.org/wiki/Diagramma\\_di\\_Voronoi](https://it.wikipedia.org/wiki/Diagramma_di_Voronoi)

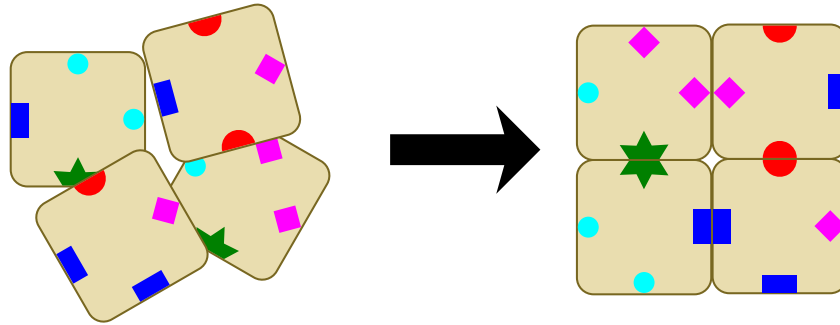




# 13. Mosaico

Devi disporre quattro carte in un quadrato in modo che due bordi che si toccano abbiano lo stesso simbolo.

Ad esempio, le seguenti quattro carte possono essere posizionate come un tale quadrato:




È possibile creare un quadrato di questo tipo con quattro delle cinque carte seguenti. Quale non si può usare?

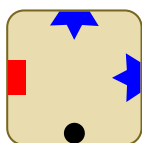
- A)
- B)
- C)
- D)
- E)




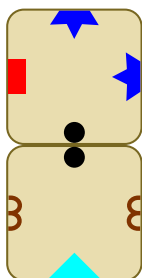
## Soluzione


Ci sono  $\binom{5}{4} \cdot 4! \cdot 4^4 = 30720$  modi diversi di scegliere e posare quattro delle cinque carte. Anche considerando che ci sono almeno quattro soluzioni dovute alla simmetria rotazionale, sono decisamente troppe per provarle tutte.

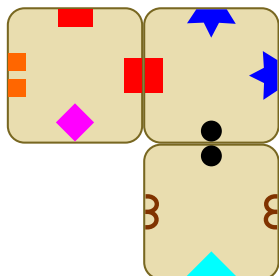
Pertanto, è opportuno osservare innanzitutto la distribuzione dei simboli sulle carte. Si noter  che la mezza stella blu  appare solo sulla carta C). Poich  appare anche due volte, cio  attraverso l'angolo, le altre due carte possono essere posizionate solo a sinistra e in basso, se non si cambia l'orientamento (che   inutile visto che   la prima carta).





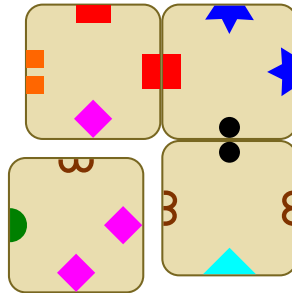
Inoltre, c'  solo un'altra carta con un cerchio nero  che deve essere posizionata sotto, cio  la carta E):



Anche per il rettangolo rosso  esiste una sola carta, cio  la carta B). Contiene rettangoli rossi in due punti, ma poich  non esiste un'altra carta con rettangoli rossi, deve essere ruotata in modo che il secondo rettangolo rosso sia in alto e non in basso. Altrimenti, si dovrebbe aggiungere una carta con un rettangolo rosso al di sotto di essa, e questa carta non esiste:

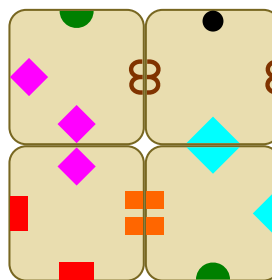


L'ultima cosa necessaria   una carta con un quadrato magenta  e due semicerchi marroni . La carta D) presenta anche questi simboli, ma nell'ordine sbagliato: i due semicerchi marroni dovrebbero venire in senso orario dopo il quadrato magenta, ma   esattamente il contrario.



Poiché non c'erano alternative per tutte le carte, si dimostra che con la carta C) con le due mezzette blu non si può posare un quadrato che soddisfi le condizioni.

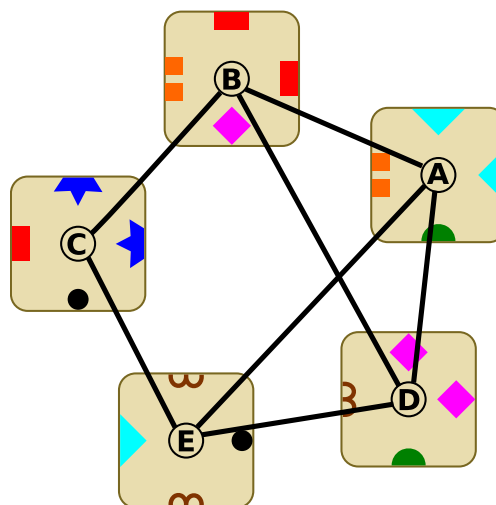
Tuttavia, tale quadrato può essere posato dalle altre carte:



Questo dimostra che la carta C) è l'unica con la quale non è possibile posare un quadrato di questo tipo e C) è la risposta corretta.

### Questa è l'informatica!

Se due carte hanno lo stesso simbolo, possono essere messe una accanto all'altra. Questo può essere rappresentato con l'aiuto di un *grafo*: le carte rappresentano i *vertici*, se c'è un simbolo comune, allora c'è un *arco*. Per le cinque carte di questo compito, il grafo si presenta come segue:



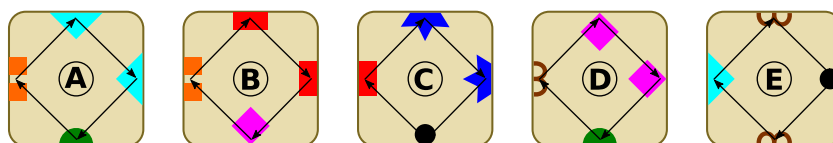
Se quattro carte sono disposte a quadrato in modo che esattamente due carte abbiano sempre lo stesso simbolo, significa che nel grafico è possibile seguire un percorso circolare con esattamente quattro



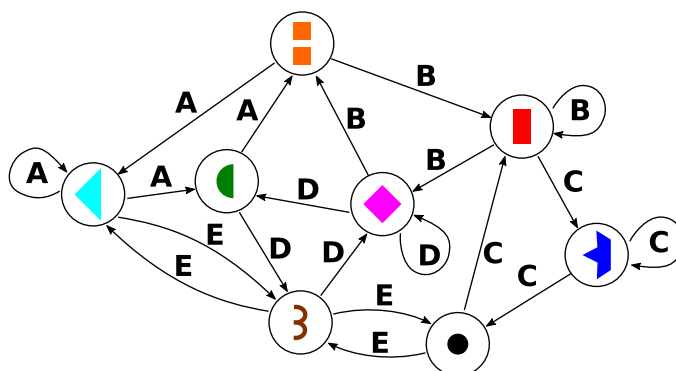
carte, per esempio da A a E a C a B e di nuovo ad A. Abbreviamo tale percorso come A-E-C-B-A e intendiamo tutti gli altri percorsi circolari che seguono questo ordine, cioè anche E-C-B-A-E, C-B-A-E-C e B-A-E-C-B.

Nel grafo ci sono esattamente tre percorsi circolari con quattro mappe: il già citato percorso circolare A-E-C-B-A, A-E-D-B-A e B-D-E-C-B. Questo riduce il numero di soluzioni possibili da  $\binom{5}{4} \cdot 4! \cdot 4^4 = 30720$  (rispettivamente 7680 se si omette la simmetria rotazionale) a 12 (rispettivamente 3 se si omette la simmetria rotazionale). Questi possono essere controllati rapidamente per trovare la soluzione corretta (A-E-D-B-A).

È anche possibile utilizzare un grafo in cui i simboli rappresentano i vertici. A tal fine, è necessario definire l'ordine dei simboli all'interno di una carta in modo uniforme, ad esempio in senso orario:



Nel grafo, un bordo direzionato esiste tra due simboli se sono adiacenti su una mappa in senso orario:



La soluzione del compito è data esattamente se si trova un percorso circolare che passa esattamente su quattro archi. Si tratta del percorso circolare - - - - , che passa sopra gli archi A, E, D e B e corrisponde quindi alla nostra soluzione.

Questo processo di concentrazione sull'essenziale e di esclusione del non importante si chiama astrazione. In questo caso, l'astrazione consente un riconoscimento molto più rapido della soluzione corretta.

### Parole chiave e siti web

- Grafo: <https://it.wikipedia.org/wiki/Grafo>
- Vertice: [https://it.wikipedia.org/wiki/Vertice\\_\(teoria\\_dei\\_grafi\)](https://it.wikipedia.org/wiki/Vertice_(teoria_dei_grafi))
- Arco: [https://it.wikipedia.org/wiki/Glossario\\_di\\_teorica\\_dei\\_grafi#Arco](https://it.wikipedia.org/wiki/Glossario_di_teorica_dei_grafi#Arco)



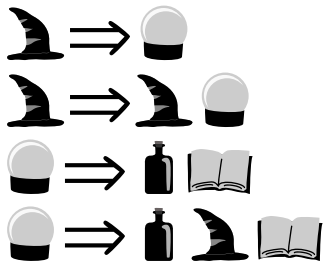
## 14. Oggetti magici

Nella terra magica ci sono quattro diversi oggetti magici:

Cappelli magici , Sfere di cristallo , Libri magici  e Pozioni magiche .
























I cappelli magici e le sfere di cristallo possono essere trasformati in due modi diversi. La tabella mostra cosa viene creato dagli oggetti in questo processo - esattamente nel posto in cui si trovavano prima e nella disposizione indicata:

da . . . nasce



Le trasformazioni possono avvenire un numero qualsiasi di volte e in qualsiasi ordine. Così, da un unico oggetto magico può nascere una lunga disposizione di oggetti.

Quale disposizione *NON* può nascere da un singolo cappello magico?

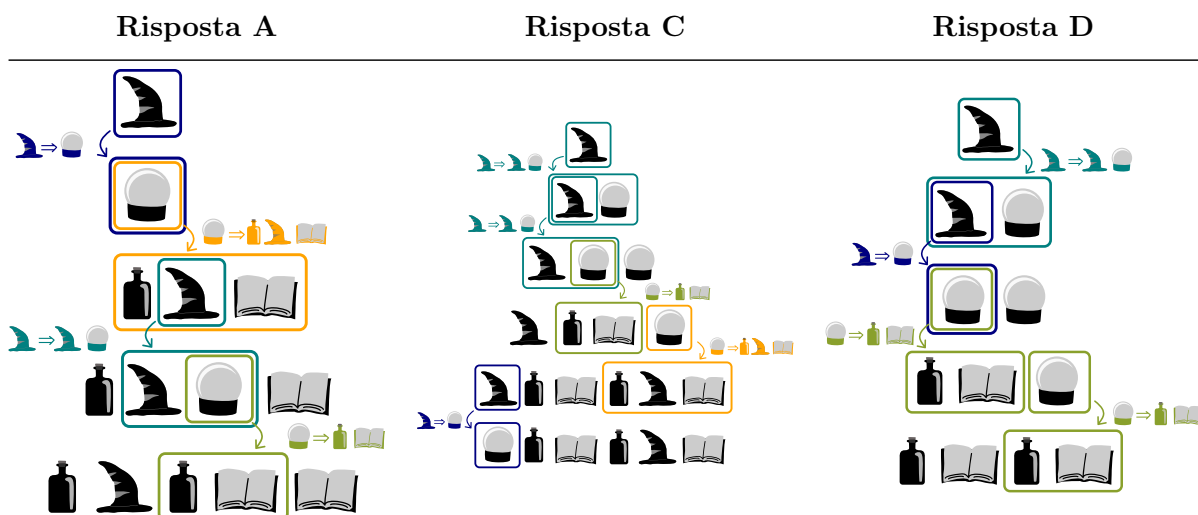
- A)     
- B)        
- C)      
- D)    



## Soluzione

La risposta B) è corretta:

Le disposizioni delle risposte A, C e D possono nascere da un unico cappello magico:



Contiene un numero diverso di libri ( $2 \times \text{book}$ ) e pozioni ( $3 \times \text{bottle}$ ). Ma ogni volta che viene creata una pozione in una trasformazione, viene creato contemporaneamente un libro ( $\text{crystal ball} \Rightarrow \text{bottle book}$  e  $\text{crystal ball} \Rightarrow \text{bottle witch hat book}$ ). Quindi, in ogni disposizione creata nella terra magica dalle trasformazioni, ci devono essere esattamente tanti libri di incantesimi quante sono le pozioni. Quindi la disposizione della risposta B) non può nascere né da un cappello magico né da una sfera di cristallo.

## Questa è l'informatica!

Quando le sfere di cristallo e i cappelli magici di questo compito del castoro vengono trasformati di volta in volta, si creano disposizioni diverse. Poiché le trasformazioni producono sempre nuove sfere di cristallo e cappelli magici, è possibile creare un numero infinito di disposizioni. Non è quindi possibile scrivere tutte le disposizioni che possono nascere con l'aiuto delle trasformazioni. Ma non è nemmeno necessario, perché l'insieme delle disposizioni è determinato proprio dalle trasformazioni stesse.

Anche prima dell'esistenza dei computer, è nata l'idea di descrivere infiniti insiemi di disposizioni con l'aiuto di un insieme relativamente piccolo e comunque finito di trasformazioni. In informatica, le trasformazioni sono chiamate *regole di sostituzione*, gli insiemi di regole sono chiamati *grammatiche* e gli insiemi che le definiscono sono di conseguenza chiamati *lingue*. Un ruolo centrale in questi *linguaggi formali* dell'informatica è svolto dal problema della decisione: una disposizione di oggetti appartiene al linguaggio (cioè: può essere creata applicando le regole) oppure no?

Nel rispondere a questo compito, hai dovuto risolvere questo problema per quattro disposizioni. Fortunatamente, la grammatica di questo compito rientra nella classe delle grammatiche *libere dal contesto*: le sfere di cristallo e i cappelli magici possono trasformarsi senza considerare ciò che li



circonda, cioè nel loro contesto. Per le grammatiche libere da contesto, il *problema della decisione* è generalmente facile da risolvere, ed è per questo che sono molto popolari in informatica, ad esempio per descrivere i linguaggi di programmazione.

## Parole chiave e siti web

- Linguaggio libero dal contesto:  
[https://it.wikipedia.org/wiki/Linguaggio\\_libero\\_dal\\_contesto](https://it.wikipedia.org/wiki/Linguaggio_libero_dal_contesto)
- Grammatica libera dal contesto:  
[https://it.wikipedia.org/wiki/Grammatica\\_libera\\_dal\\_contesto](https://it.wikipedia.org/wiki/Grammatica_libera_dal_contesto)
- Linguaggio formale: [https://it.wikipedia.org/wiki/Linguaggio\\_formale](https://it.wikipedia.org/wiki/Linguaggio_formale)
- Decidibilità: [https://it.wikipedia.org/wiki/Decidibilità](https://it.wikipedia.org/wiki/Decidibilit%C3%A0)













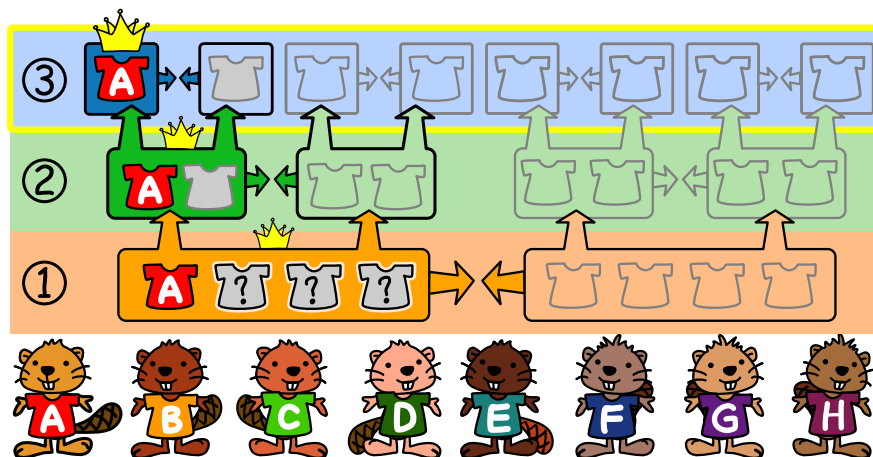
# 15. Campionato dei Castori

8 castori partecipano al Campionato dei castori. Ci sono tre turni. In ogni turno, ogni castoro raccoglie punti.

- Turno ①: si formano 2 squadre casuali di 4 castori ciascuna. I punti dei singoli castori vengono sommati. La squadra con il maggior numero di punti vince e passa al turno ②. I perdenti continuano a giocare e si accordano tra loro per i posti dal 5 all'8.
- Turno ②: Questo viene condotto con le stesse regole. Le squadre sono ora composte da 2 castori. I vincitori passano alla finale. I perdenti continuano a giocare e a stabilire i posti d'onore tra di loro.
- Turno ③: La finale! Non ci sono squadre, ma 2 castori singoli che si sfidano tra loro.

La castorina Ada è la vincitrice del campionato dei castori. Di seguito sono riportati i punti ottenuti da ciascun castoro in ogni turno.




								
Nome	Ada	Brown	Candy	Daisy	Eden	Funny	George	Hugh
①	15	16	19	18	17	20	19	19
②	20	27	30	24	28	24	30	30
③	10	14	11	15	16	13	9	12



Quali tre castori facevano parte della squadra di Ada nel primo turno?



## Soluzione

I tre compagni di squadra di Ada nel turno ① erano Daisy , Funny  e George .

La finale sarà giocata individualmente. George è l'unico castoro con meno punti di Ada. Nel turno ② devono quindi aver fatto parte della stessa squadra.

Nel turno ② hanno raggiunto insieme 50 punti. Questo valore deve essere superiore al punteggio totale dell'altra squadra di due persone. I due castori Daisy e Funny sono l'unica coppia il cui totale dei punti è inferiore a 50. Pertanto, devono aver fatto parte della stessa squadra di Ada e George nel turno ①.

Poiché ora abbiamo trovato tutti e quattro i castori della squadra di Ada nel turno ①, conosciamo anche la composizione della seconda squadra nel turno ①.

Nel turno ① la squadra (Ada, Daisy, Funny, George) ha totalizzato 72 punti. L'altra squadra (Brown, Candy, Eden, Hugh) ha totalizzato solo 71 punti. La squadra di Ada ha vinto.

Nel turno ②, (Ada, George) ha totalizzato 50 punti, mentre (Daisy, Funny) ha ottenuto solo 48 punti. Nel round ③, cioè la finale, Ada vince con 10 punti contro 9 contro George. Ada vince il campionato dei castori.

## Questa è l'informatica!

Per risolvere questo compito, possiamo formare sistematicamente tutte le squadre possibili al primo turno. Se conosciamo una delle due squadre, anche la seconda è nota. In totale, esistono  $\binom{7}{3} = 35$  combinazioni. Per ognuna di queste combinazioni, dobbiamo esaminare i risultati nel turno ①, nel turno ② e nella finale prima di poter decidere quali sono stati effettivamente i compagni di squadra di Ada nel primo turno. Questo richiede molto tempo.

Per risolvere un compito come questo, gli informatici cercano approcci più efficienti. Invece di procedere in avanti, cioè dal primo al terzo round, si può anche ricavare la soluzione corretta a ritroso. Questo può essere fatto anche molto rapidamente, come abbiamo già visto nella spiegazione precedente.

Questo metodo è chiamato *ricerca a ritroso*. Si utilizza in situazioni in cui si cerca una soluzione che deve soddisfare determinate condizioni. In alcuni casi, per trovare una soluzione si combinano una ricerca *in avanti* e una *all'indietro*.


La ricerca in avanti e la ricerca all'indietro sono due strategie generali di risoluzione dei problemi. Vengono utilizzati per risolvere problemi in tutte le discipline, non solo in informatica.

## Parole chiave e siti web

- Algoritmo di Dijkstra: [https://it.wikipedia.org/wiki/Algoritmo\\_di\\_Dijkstra](https://it.wikipedia.org/wiki/Algoritmo_di_Dijkstra)
- Algoritmo A\*: [https://it.wikipedia.org/wiki/Algoritmo\\_A\\*](https://it.wikipedia.org/wiki/Algoritmo_A*)



## A. Autori dei quesiti

 Esraa Almajhad


 Wilfried Baumann

 Linda Björk Bergsveinsdóttir

 Graeme Buckie


 Sarah Chan

 Kris Coolsaet

 Darija Dasović

 Christian Datzko

 Susanne Datzko


 Justina Dauksaite

 Gerald Futschek


 Christian Giang

 Alisher Ikramov

 Thomas Ioannou

 Sangsu Jeong

 Mile Jovanov

 Dong Yoon Kim

 Hakin Kim

 Jihye Kim

 Regula Lacher

 Taina Lehtimäki


 Monika Maneva


 Zoran Milevski

 Madhavan Mukund

 Ágnes Erdősné Németh

 Ilze Nilandere

 Veronika Ognjanovska

 Mārtiņš Opmanis

 Elsa Pellet

 Margot Phillipps

 Zsuzsa Pluhár

 Wolfgang Pohl


 John-Paul Pretti

 Lorenzo Repetto

 Chris Roffey

 Kirsten Schlüter

 Giovanni Serafini

 Timur Sitdikov


 Bernadette Spieler

 Emil Stankov

 Veronika Stefanovska

 Goran Sukovic


 Jiří Vaníček

 Willem van der Vegt

 Rechilda Villame

 Florentina Voboril

 Michael Weigend

 Kyra Willekes

 Hongjin Yeh



## B. Sponsoring: concorso 2022

**HASLERSTIFTUNG**

<http://www.haslerstiftung.ch/>



Kanton Zürich  
Volkswirtschaftsdirektion  
Amt für Wirtschaft und Arbeit

Standortförderung beim Amt für Wirtschaft und Arbeit Kanton Zürich



<http://www.ubs.com/>



<http://www.verkehrshaus.ch/>

Musée des transports, Lucerne



i-factory (Musée des transports, Lucerne)



<http://senarclens.com/>

Senarclens Leu & Partner



AUSBILDUNGS- UND BERATUNGSZENTRUM  
FÜR INFORMATIKUNTERRICHT

<http://www.abz.inf.ethz.ch/>

Ausbildungs- und Beratungszentrum für Informatikunterricht der ETH Zürich.



<http://www.hepl.ch/>

Haute école pédagogique du canton de Vaud

Scuola universitaria professionale  
della Svizzera italiana

<http://www.supsi.ch/home/supsi.html>

La Scuola universitaria professionale della Svizzera italiana (SUPSI)

**SUPSI**



## C. Ulteriori offerte



La Fiamma IT: <https://it-feuer.ch/it/>

In Svizzera, numerose organizzazioni si impegnano per la formazione delle giovani leve nell'ambito dell'informatica. L'iniziativa «La Fiamma IT» vuole unire queste forze e contribuire insieme a diffondere il tema nell'opinione pubblica in tutta la Svizzera. La fiamma IT presenta numerose offerte rivolte sia ai docenti che agli studenti.



CoetryLab: <https://www.coetry-lab.org/>

Il team del CoetryLab (Zürich) vuole dare ai bambini e ai giovani l'accesso alla programmazione e ai media. Il Coetry-Lab vuole essere il luogo di sperimentazione e progettazione extrascolastica e aprire il mondo del coding a tutti. Le loro idee possono essere realizzate in modo creativo e siti web, applicazioni, giochi e molto altro possono essere sviluppati in team o da soli.



Roteco: <https://www.roteco.ch/it/>

Il progetto Roteco consiste in una comunità di insegnanti desiderosi di preparare gli allievi per la società digitale. In questa comunità gli insegnanti trovano, sviluppano e si scambiano attività didattiche inerenti la robotica educativa e più in generale le scienze informatiche pronte da essere utilizzate in classe e vengono informati con le ultime novità e corsi in questi campi.

010100110101011001001001  
010000010010110101010011  
010100110100100101000101  
001011010101001101010011  
010010010100100100100001

**SSII**

[www.svia-ssie-ssii.ch](http://www.svia-ssie-ssii.ch)  
schweizerischervereinfürinformatikind  
erausbildung//sociétésuissepourl'infor  
matique dans l'enseignement//societàsviz  
zera per l'informaticanell'insegnamento

Diventate membri della SSII <http://svia-ssie-ssii.ch/verein/mitgliedschaft/> sostenendo in questo modo il Castoro Informatico.

Chi insegna presso una scuola dell'obbligo, media superiore, professionale o universitaria in Svizzera può diventare membro ordinario della SSII.

Scuole, associazioni o altre organizzazioni possono essere ammesse come membro collettivo.