INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Quesiti e soluzioni 2022

7º e 8º anno scolastico

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

A cura di:

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

SSII
www.svia-ssie-ssii.ch
schweizerischer verein für informatik in der ausbildung // société suisse pour l'informatique dans l'enseignement // società svizzera per l'informatica nell'insegnamento

HASLERSTIFTUNG ETH zürich | ABZ AUSBILDUNGS- UND BERATUNGSZENTRUM FÜR INFORMATIKUNTERRICHT hep/ Scuola universitaria professionale della Svizzera italiana SUPSI

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, Ausbildungges- und Beratungszentrum für Informatikunterricht
Tobias Berner: Pädagogische Hoschschule 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 Šiaulys, 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. Ringraziamo per la buona collaborazione:
Eljakim Schrijvers, Justina Dauksaite, Dave Oostendorp, Alieke Stijf, Kyra Willekes, Jo-Ann Bolten: 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 22 novembre 2023 con il sistema per la preparazione di testi 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.

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.bebras.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» (3º e 4º 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:

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

Ogni categoria aveva quesiti classificati in tre livelli di difficoltà: facile, medio e difficile. Alla categoria del 3º e 4º anno scolastico sono stati assegnati 9 quesiti da risolvere, di cui 3 facili, 3 medi e 3 difficili. Alla categoria del 5º e 6º 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, 5º e 6º anno scolastico: 36 punti).

Il punteggio massimo totalizzabile era dunque pari a 180 punti (risp., Piccolo castoro: 108 punti, 5º e 6º 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/`

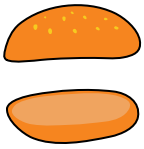
Indice

1. Ricetta hamburger

La castorina Jess prepara gli hamburger. Per farli segue tre regole:

1. la salsa è direttamente sulla carne.
2. la carne e il formaggio sono sotto i cetrioli, la lattuga e le cipolle.
3. le cipolle non toccano il panino.

Ingredienti dell'hamburger:

Panino	**Carne**	**Salsa**	**Cetrioli**	**Lattuga**	**Cipolle**	**Formaggio**
						

Quale hamburger è composto secondo le tre regole?

A) 

B) 

C) 

D) 

Soluzione

La risposta corretta è D. 

Per trovare la soluzione, è necessario controllare ogni hamburger per vedere se è messo insieme in modo da seguire tutte e tre le regole.

A) Questo hamburger segue le regole 1 e 2. Ma le cipolle toccano il panino, quindi non rispetta la regola 3.

B) Questo hamburger segue la regola 1. Ma la lattuga è sotto la carne e il formaggio, quindi la regola 2 non è stata rispettata.

C) Questo hamburger segue la regola 2 perché la carne e il formaggio sono sotto i cetrioli, la lattuga e le cipolle. Inoltre, questo hamburger di castoro segue la regola 3 perché le cipolle non toccano il panino. Tuttavia, la salsa non viene versata direttamente sulla carne. Pertanto, la regola 1 non è stata rispettata.

D) Questo hamburger soddisfa tutte le regole.

Questa è l'informatica!

Gli hamburger in questo compito sono realizzati secondo tre regole. Per ogni hamburger che prepara, la castorina Jess deve seguire ognuna delle tre regole. Se non rispetta una sola delle regole, l'hamburger non è giusto. Ognuna delle tre regole è una condizione che deve essere soddisfatta affinché ogni hamburger sia giusto.

In informatica, il controllo dei vincoli è spesso usato per scoprire se una soluzione segue tutte le regole date. In questo controllo, si collegano tutte le regole (condizioni) con l'operatore *E*. Ciò significa che tutte le regole (condizioni) devono essere soddisfatte contemporaneamente.

Verificare se una determinata soluzione soddisfa tutti i vincoli è un compito fondamentalmente diverso dal trovare una possibile soluzione. Si tratta del cosiddetto *problema di soddisfacimento di vincoli*. Nella maggior parte dei casi, è molto più difficile trovare una soluzione che soddisfi tutti i vincoli che verificare se una soluzione soddisfa tutti i vincoli. Questo vale anche per un computer.

Parole chiave e siti web

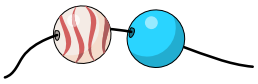
- Programmazione a vincoli: `https://it.wikipedia.org/wiki/Programmazione_a_vincoli`
- Problema di soddisfacimento di vincoli: `https://it.wikipedia.org/wiki/Problema_di_soddisfacimento_di_vincoli`
- Congiunzione logica: `https://it.wikipedia.org/wiki/Congiunzione_logica`

- NP: `https://it.wikipedia.org/wiki/NP_(complessità)`

2. Collana da marinaio

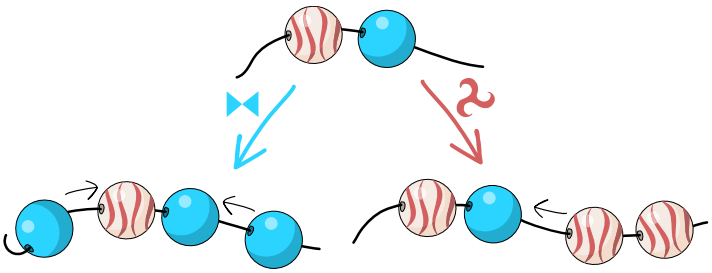
Ecco il manuale per la collana da marinaio di Monika con perline a onda bianche e rosse e perline blu semplici.

Inizia sempre con una perlina a onda e una perlina blu in questo ordine:



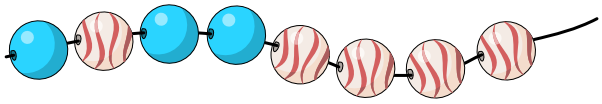
Poi puoi estendere la collana da marinaio,

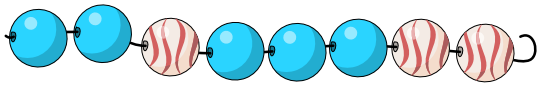
- aggiungendo una perlina blu a ciascuna estremità della stringa (⋈)
- oppure aggiungendo due perline a onda all'estremità destra della stringa (ᔓ)

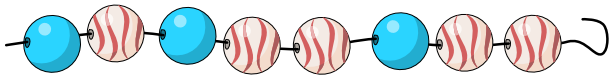


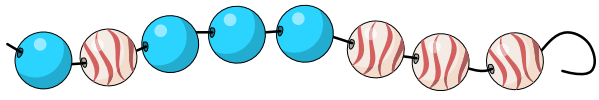
Puoi eseguire queste azioni più volte per creare collane sempre più lunghe.

Quale delle seguenti collane ***non*** *è una delle collane da marinaio di Monika?*

A) 

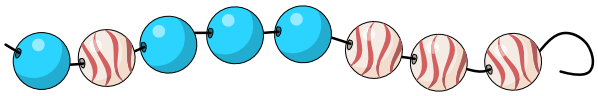
B) 

C) 

D) 

Soluzione

D è la risposta corretta.



Puoi risolvere il problema in diversi modi.

Per esempio, trovando prima le due perline iniziali di ogni collana ed eseguendo poi una serie di azioni ⋈ e ⋈.

- Nella collana A, puoi iniziare con la seconda e la terza perlina e poi eseguire le azioni ⋈ - ∿ - ∿.
- Per la collana B, puoi iniziare con la terza e la quarta perlina e poi eseguire le azioni ⋈ - ⋈ - ∿.
- Per la collana C, puoi iniziare con la seconda e la terza perlina e poi eseguire le azioni ∿ - ⋈ - ∿.
- Tuttavia, se guardi la collana D, la seconda e la terza perlina devono essere l'inizio. L'azione B può essere eseguita una volta, ma dopo di essa non ci sono altre azioni per ottenere il resto della catena.

Questo approccio non funziona bene se la collana è molto lunga e ha molte possibili perline di partenza. In questo caso, un approccio decostruttivo potrebbe funzionare meglio. In questo caso rimuovi ripetutamente le perline eseguendo l'azione B o l'azione W al contrario, finché non rimangono solo due perline.

Una terza strategia si avvale della *parità*. Secondo le istruzioni della collana del marinaio, c'è sempre un numero dispari di perline blu e un numero dispari di perline ondulate rosse e bianche («parità dispari»). Capisci perché?

La collana D ha un numero pari di entrambi i tipi di perline e quindi non può essere una delle collane da marinaio di Monika.

Questa è l'informatica!

In questa attività puoi infilare le perline solo alle estremità della collana. Non puoi inserire una perlina al centro. Inoltre, non puoi rimuovere una perlina dal centro senza aver prima sfilato le perline dall'estremità della collana.

Questo tipo di struttura di memoria, in cui è possibile aggiungere e rimuovere facilmente elementi alle estremità ma non al centro, è chiamata in informatica *coda a doppia estremità* o *coda deque* (deque si pronuncia come «deck»).

Le code deque possono essere utilizzate per memorizzare la cronologia del browser, per programmare i lavori di stampa e anche per verificare la validità delle espressioni matematiche. Ad esempio, il

controllo della corrispondenza delle parentesi può essere fatto più o meno nello stesso modo in cui si controlla se una collana è una delle collane di marinaio di Monika.

Parole chiave e siti web

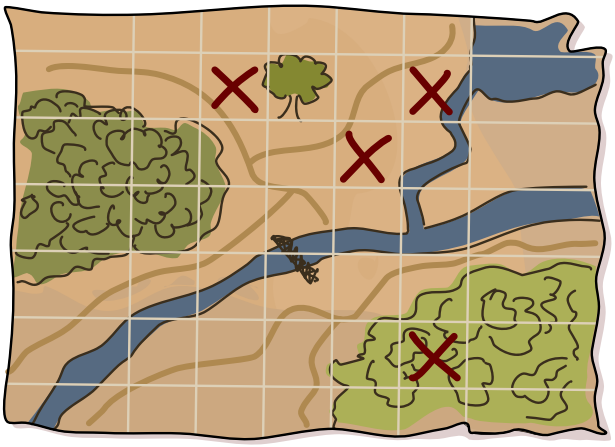
- double-ended queue: `https://it.wikipedia.org/wiki/Deque`

3. Mappa del tesoro

Il castoro Bilbo ha due buoni nascondigli per il suo cibo. Su una mappa segna i due campi dove si trovano i nascondigli con ✕. Ma cosa succede se altri castori trovano la mappa e quindi i nascondigli?

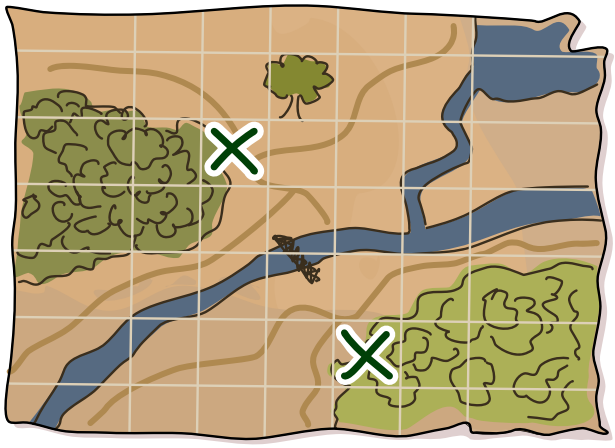
Per confondere le cose, Bilbo segna altri campi con ✕. Lo fa in modo che in ogni riga e colonna della mappa sia segnato un numero pari di caselle. Poi rimuove i due ✕ dai campi con i suoi nascondigli. Di seguito è possibile vedere il risultato.

In quali campi si trovano i nascondigli di Bilbo?

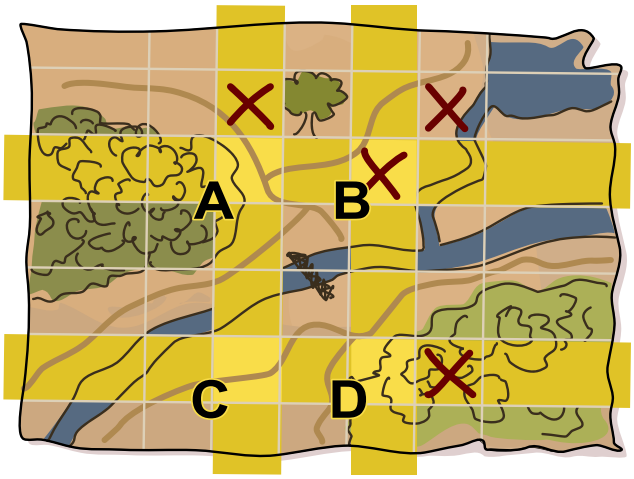


Soluzione

Ecco i due nascondigli:



Per trovarli, osserviamo la mappa originale e notiamo che ci sono due righe e due colonne in cui il numero di ✕ non è pari: le righe 3 e 6 e le colonne 3 e 5.



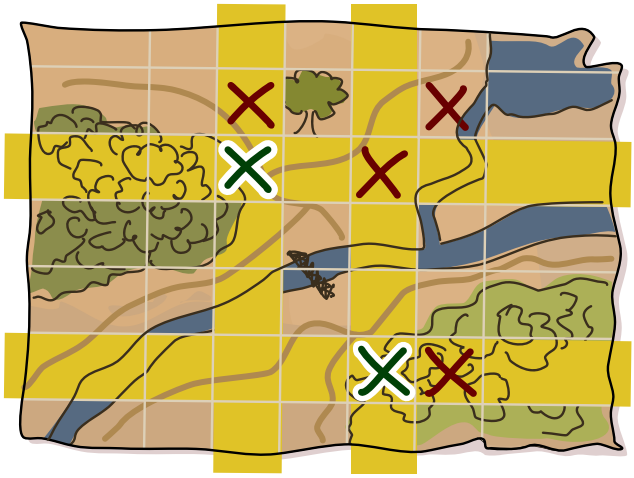
Dopo tutto, i ✕ che segnalano i nascondigli sono stati rimossi. Sappiamo che deve esserci un numero pari di ✕ in tutte le righe e le colonne dopo che le ✕ cancellate sono state reinserite.

Le righe e le colonne interessate si sovrappongono e hanno quattro campi comuni (A, B, C e D). Questi «campi intersecanti» sono di particolare interesse per noi. Se contrassegniamo i campi al di fuori di un campo di intersezione con ✕, possiamo ottenere un numero ✕ pari in una colonna, mentre il numero nella rispettiva riga diventa dispari e viceversa. Pertanto, le ✕ dei due nascondigli devono trovarsi sui campi di intersezione.

Il campo di intersezione B è già contrassegnato da una ✕: non può essere un nascondiglio perché sappiamo che Bilbo ha cancellato la ✕ dei nascondigli.

Quindi, per restituire un numero pari di ✕ nella riga 3, dobbiamo contrassegnare l'intersezione A con una ✕. Lì c'è un nascondiglio. L'altro nascondiglio non può trovarsi nell'intersezione C, perché

allora ci sarebbero tre ✕ in quella colonna. Quindi l'altro nascondiglio si trova all'intersezione D. Ecco la mappa prima che Bilbo cancellasse le ✕, con un numero pari di ✕ in ogni riga e colonna:



Questa è l'informatica!

Bilbo utilizza un trucco spesso usato in informatica: i *bit di parità*. Fanno parte di un insieme di tecniche di *rilevazione e correzione d'errore*. L'idea è che ogni volta che memorizziamo o trasmettiamo dati come una serie di *bits* (che possono essere 0 o 1), aggiungiamo bit supplementari per aiutarci a rilevare se si sono prodotti errori di trasmissione o di memorizzazione, in genere quando un bit è stato distorto, cioè quando un bit è stato inviato come 1 e ricevuto erroneamente come 0, o viceversa.

Ad esempio, se utilizziamo un semplice codice di rilevamento degli errori, viene aggiunto un bit di parità in modo che il numero di uno sia sempre pari. 0110101 viene aggiunto uno 0 per diventare 01101010 (il numero di bit a «1» rimane pari). Se il secondo bit è stato invertito e il messaggio viene ora inviato a 00101010, il messaggio ricevuto non soddisfa il requisito di parità (tre bit sono bit a «1»). È importante notare che questo metodo non è in grado di rilevare un problema se più di un bit è in errore.

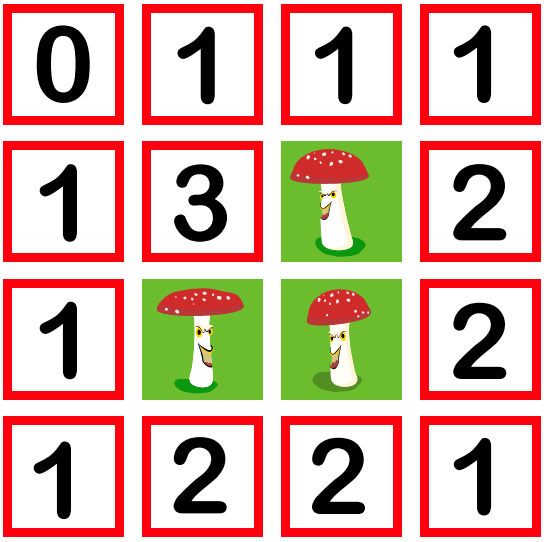
Parole chiave e siti web

- Bit: `https://it.wikipedia.org/wiki/Bit`
- Bit di parità: `https://it.wikipedia.org/wiki/Bit_di_parità`
- Rilevazione e correzione d'errore: `https://it.wikipedia.org/wiki/Rilevazione_e_correzione_d'errore`

4. Attenzione ai funghi

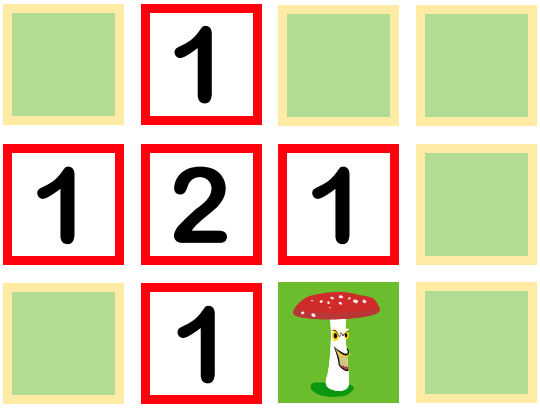
Nel gioco «Attenzione ai funghi», all'inizio è visibile esattamente un fungo. Tutte le altre caselle del tabellone sono coperte. Se si scopre un campo, appare un altro fungo o il numero di funghi sui campi vicini. Se si scoprono tutte le caselle in cui non è nascosto alcun fungo, si vince.

Ecco un esempio di una tavola completamente scoperta:



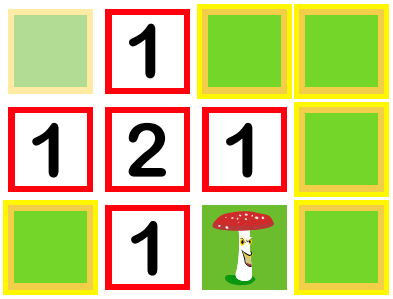
Hai iniziato una nuova partita e hai già scoperto alcune caselle.

Su quale dei campi rimanenti non c'è sicuramente un fungo?

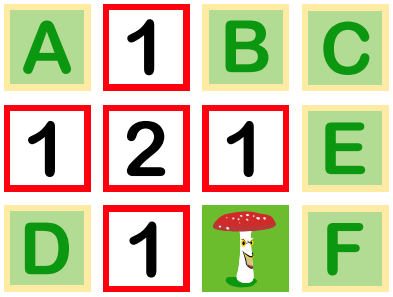


Soluzione

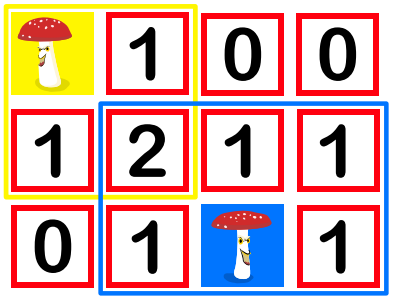
Questa è la soluzione:



Per spiegare la risposta corretta, etichettiamo i quadrati coperti con delle lettere. Inoltre, diciamo che un numero N su un campo è «esaurito» se c'è già un fungo scoperto su ciascuno degli N campi vicini di questo numero; non ci possono quindi essere altri funghi su altri campi vicini.



- Non c'è nessun fungo sulla casella D perché il numero 1 alla sua destra è esaurito.
- Nei campi B, C, E e F non c'è nessun fungo, perché il numero 1 di questi campi, comunemente vicino, è esaurito.
- C'è un fungo sul campo A, perché altrimenti i numeri 1, 2 e 1 non indicherebbero correttamente il numero di funghi sui campi vicini.



Quindi c'è un fungo nascosto nel campo A. I campi B, C, D, E e F possono essere scoperti.

Questa è l'informatica!

Come abbiamo proceduto? A volte è necessario partire da un'ipotesi per poi ragionare in modo logico. Se si trova una contraddizione, si torna indietro e si prosegue l'ipotesi seguente più plausibile. Si tratta di una ricerca «mirata» e non di tentativi ed errori.

Usando un computer come si potrebbe risolvere questo problema? Se si scopre almeno un campo con un rospo, si possono derivare delle semplici regole. Ad esempio, se il campo con il numero 1 copre già un campo vicino con un rospo scoperto, non può esserci un altro rospo lì vicino. Se queste regole sono formulate con precisione per ogni numero, un computer potrebbe eseguirle passo dopo passo come *istruzioni*. In definitiva, avremmo un *algoritmo* che sarebbe necessario eseguire per avere successo nel gioco (con almeno un rospo scoperto).

Parole chiave e siti web

- Campo minato: `https://it.wikipedia.org/wiki/Campo_minato_(videogioco)`
- Algoritmo: `https://it.wikipedia.org/wiki/Algoritmo`

5. Ricamo

Lana ha una macchina da ricamo programmabile. La macchina può ricamare due tipi di motivi:  o  . Per creare questo motivo composto  sono necessari entrambi i motivi  e  . Nel mezzo, il tessuto deve essere spinto indietro di un punto.

Lana può programmare la macchina da ricamo con i seguenti tre pulsanti:

 La macchina da ricamo ricama  .

 La macchina da ricamo ricama  .

 Il tessuto viene spostato indietro di un punto.

Un programma viene creato con i tasti ed eseguito ripetutamente dalla macchina da ricamo.

Ad esempio, la macchina da ricamo creerà...

- ... con questo programma  ...
- ... questo motivo: 

Quale dei seguenti programmi ha utilizzato Lana per creare questo motivo?

A) 

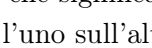
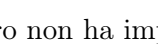
B) 

C) 

D) 

Soluzione

La risposta corretta è C). 

Per determinare il programma di Lana, cerchiamo innanzitutto la parte dello schema che si ripete:  . I primi 2 punti devono essere un  . Per questo utilizza  . All'inizio del programma di Lana devono esserci 2  . Il programma D non è corretto perché inizia con un  . Il punto successivo dello schema è una stella  . Per ricamare una stella la macchina deve cucire  e  uno sopra l'altro, il che significa che il tessuto deve essere spostato nel mezzo. L'ordine in cui  e  sono ricamati l'uno sull'altro non ha importanza. A questo scopo puoi utilizzare le due seguenti varianti di programma:  o  .

I quattro programmi producono i seguenti schemi:

	programma	schema generato
A		
B		
C		
D		

Nel programma B e D i punti non sono nell'ordine corretto. I programmi A e C sono uguali fino al quinto punto ricamato. Il Programma A aggiunge altre due croci dietro la seconda stella. Quindi, quando il programma A viene ripetuto, ci sono quattro croci tra la seconda stella e quella successiva, invece di due sole croci.

Ecco perché solo il programma C è corretto.

Questa è l'informatica!

In questo compito, uno schema ricorrente viene creato da una sequenza di istruzioni. Anche nell'informatica, i problemi più grandi e complicati vengono spesso suddivisi in problemi più piccoli, più facili da comprendere, risolvere e, ad esempio, programmare. Un'abilità importante in questo processo è riconoscere queste sequenze di modelli ricorrenti per riutilizzare una soluzione. Questo può essere fatto, ad esempio, sotto forma di *cicli*.

Il programma generato dalla macchina da ricamo è un elenco di istruzioni scritte in un linguaggio di programmazione. In sostanza, una macchina da ricamo programmabile non è altro che un robot o un computer che esegue istruzioni. Proprio come una macchina da ricamo ricama i punti esatti, un computer esegue le istruzioni esatte di un programma. Seguire esattamente le istruzioni è un concetto importante nell'informatica. L'ordine delle istruzioni è altrettanto importante. Se cambiamo l'ordine, di solito cambia anche l'output del programma. Nel nostro caso, questo significa che una

sequenza diversa di istruzioni darà luogo a una sequenza diversa di punti e quindi a un disegno diverso (eccezione: stiamo ricamando una stella).

Parole chiave e siti web

- Linguaggio di programmazione: `https://it.wikipedia.org/wiki/Linguaggio_di_programmazione`
- Algoritmo: `https://it.wikipedia.org/wiki/Algoritmo`

6. Bulloni e dadi

Ben è alla catena di montaggio e lavora i componenti: dadi e bulloni.

Ben segue rigorosamente la seguente procedura:

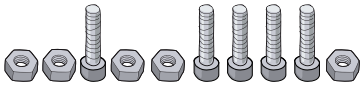
- Ben prende il componente successivo dalla catena di montaggio.
- Quando Ben ha preso un dado dalla catena di montaggio, lo mette nel secchio.
- Quando Ben ha preso un bullone dalla catena di montaggio, prende un dado dal secchio, lo avvita sul bullone e mette il pezzo finito nella scatola.

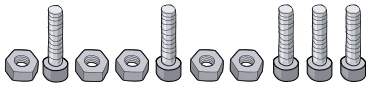
In questa procedura possono verificarsi due errori:

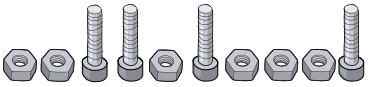
1. Ben prende un bullone dalla catena di montaggio, ma nel secchio non c'è nessun dado da avvitare.
2. Ben ha lavorato tutti i componenti della catena di montaggio, ma ci sono ancora dadi nel secchio.

Il secchio per i dadi è sufficientemente grande e vuoto all'inizio. Quale delle sequenze di dadi e bulloni può essere elaborata da Ben da sinistra a destra senza commettere errori?

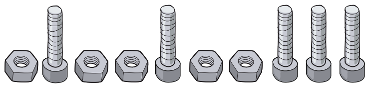
A) 

B) 

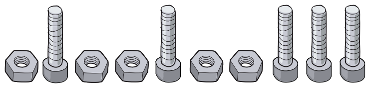
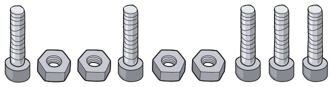
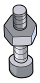
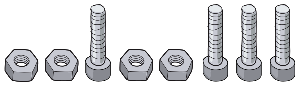
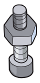
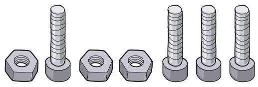
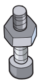
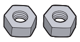
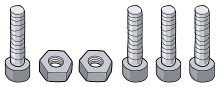
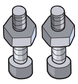
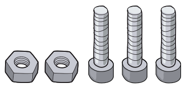
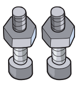
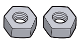
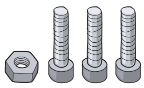
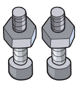
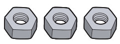
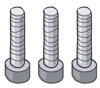
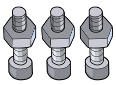
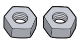
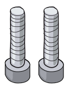
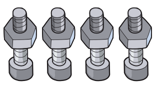
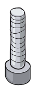
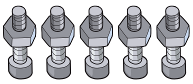
C) 

D) 

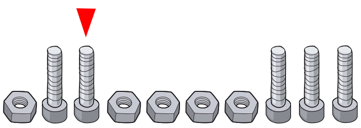
Soluzione

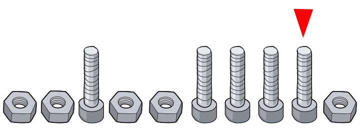
La risposta corretta è C) 

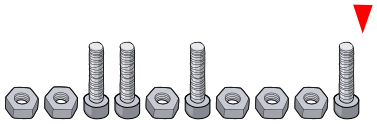
La tabella mostra lo stato della scatola per i pezzi finiti, del secchio per i dadi e della catena di montaggio.

Scatola	Secchio	Catena di montaggio
vuota	*vuoto*	
vuota		
	*vuoto*	
		
		
		
		
		
		
		
	*vuoto*	*vuota*

Perché le altre risposte sono sbagliate?

A)  porta a un errore nella posizione contrassegnata. Poi Ben ha preso un bullone, ma nel secchio non c'è più il dado.

B)  porta a un errore nella posizione contrassegnata. Finora Ben ha avvitato 4 dadi su quattro bullone. Quindi il secchio è vuoto. Ma ora ha preso un quinto bullone per il quale non ha più un dado.

D)  porta a un errore dopo l'elaborazione dell'intera sequenza. Questo perché sono stati avvitati 4 dadi su 4 bulloni e sono rimasti 2 dadi.

Questa è l'informatica!

Ben elabora i componenti che vengono consegnati uno per uno dalla catena di montaggio. Nel processo, utilizza un grande secchio per conservare temporaneamente i dadi. Una disposizione simile viene utilizzata in *informatica teorica* come modello per gli *algoritmi* in grado di risolvere una certa classe di problemi: automi a pila.

Un automa a pila elabora i dati (numeri o caratteri) che riceve in ingresso uno per uno. Ha un'unica memoria infinita, una pila. A differenza del secchio nel compito, gli elementi della pila hanno un certo ordine e si può togliere da una cantina solo l'elemento che si è messo per ultimo («last in first out», LIFO). Un automa di pila può essere utilizzato per riconoscere un *linguaggio libero dal contesto*.

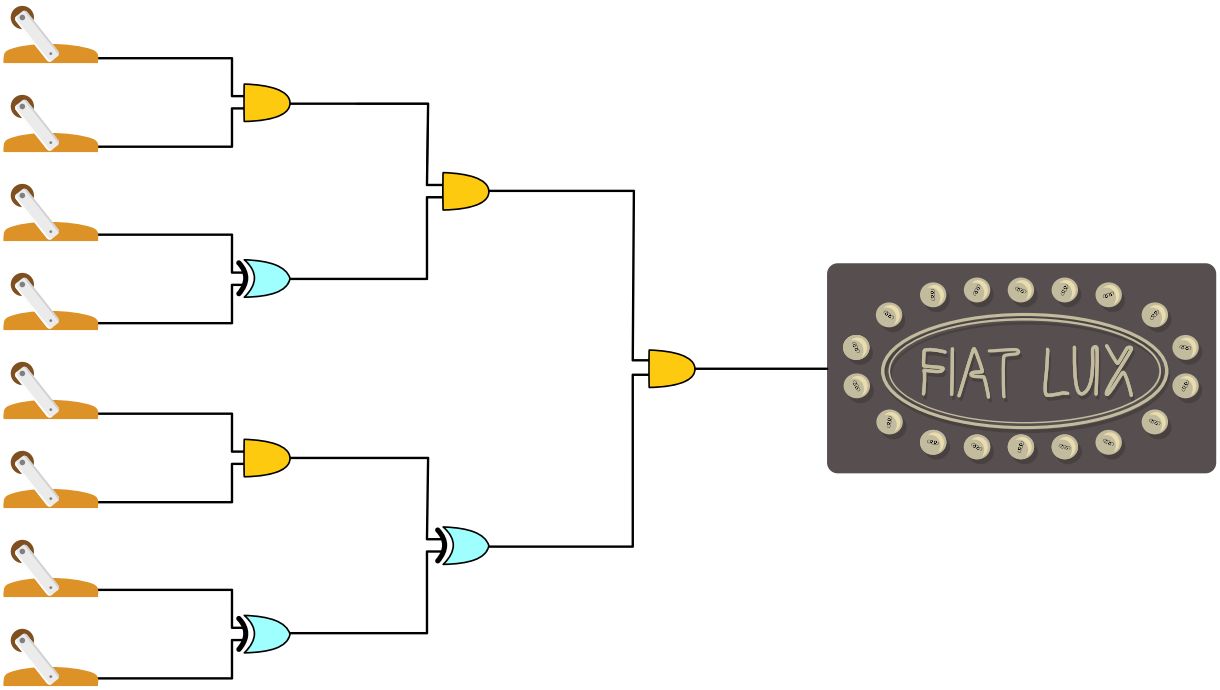
In informatica, un linguaggio è un insieme di stringhe formate secondo determinate regole. Un tipo semplice di linguaggio è il linguaggio libero dal contesto. Un esempio di linguaggio libero dal contesto è costituito da tutte le espressioni ben formate di parentesi. In un'espressione ben formata, ogni parentesi aperta viene richiusa. Le espressioni ben formate sono, ad esempio, ((( ))) e (()()). Non ben formati, invece, sono ((() e ())(. Si può pensare ai dadi e ai bulloni del compito come a delle parentesi di apertura e chiusura. Quindi Ben elabora una sequenza di componenti sulla catena di montaggio senza errori solo se rappresenta un'espressione di parentesi ben formata. La verifica delle espressioni di parentesi è un compito importante di un compilatore che traduce i testi dei programmi in programmi eseguibili. Questo perché le chiamate di funzione annidate e le espressioni aritmetiche con parentesi sono presenti nei testi dei programmi della maggior parte dei linguaggi di programmazione.

Parole chiave e siti web

- Informatica teorica: `https://it.wikipedia.org/wiki/Informatica_teorica`
- Automa a pila: `https://it.wikipedia.org/wiki/Automa_a_pila`
- Linguaggio libero dal contesto: `https://it.wikipedia.org/wiki/Linguaggio_libero_dal_contesto`

7. FIAT LUX!

Il gioco «FIAT LUX!» ha 8 interruttori che possono essere attivati o disattivati. Da questi interruttori, i fili passano attraverso alcuni componenti e infine a un'insegna al neon.

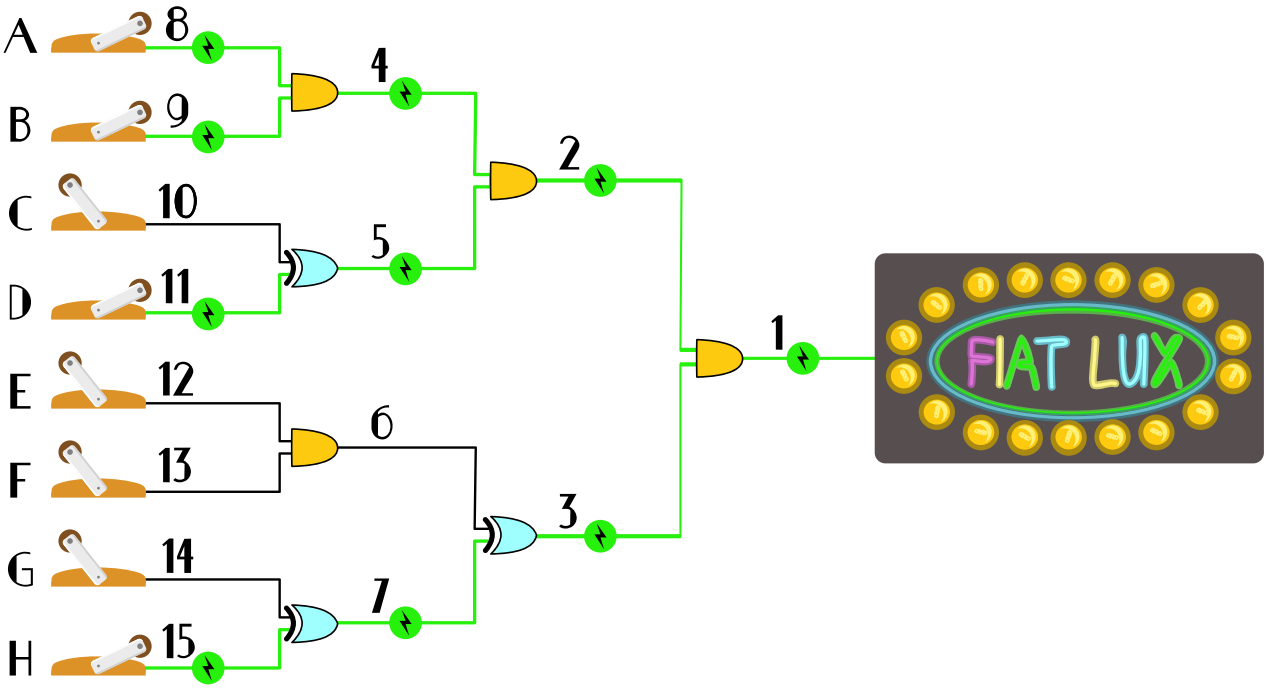


L'uscita del componente  è attiva solo quando entrambi i fili in ingresso sono attivi. L'uscita del componente  è attiva quando è attivo esattamente uno dei fili in ingresso.

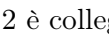
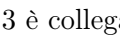
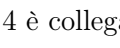
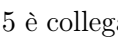
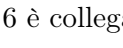
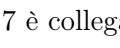
Quali interruttori  *devono essere attivati per accendere l'insegna al neon?*

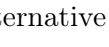
Soluzione

Una possibile soluzione è la seguente:



È possibile trovarla facilmente risolvendo il problema da dietro. Il filo collegato 1 è collegato a un componente  . Affinché l'uscita sia _on, entrambi i fili in ingresso 2 e 3 devono essere *ON*.

- Il filo 2 è collegato a un componente  . Affinché l'uscita sia *ON*, entrambi i fili 4 e 5 in ingresso devono essere *ON*.
- Il filo 3 è collegato a un componente  . Affinché l'uscita sia *ON*, esattamente uno dei due fili in ingresso deve essere *ON*, ad esempio il filo 7. Quindi il filo 6 deve essere *OFF*.
- Il filo 4 è collegato a un componente  . Affinché l'uscita sia *ON*, entrambi i fili in ingresso 8 e 9 devono essere *ON*, quindi anche gli interruttori A e B devono essere *ON*:  .
- Il filo 5 è collegato a un componente  . Affinché questo sia *ON* all'uscita, esattamente uno dei due fili in ingresso deve essere *ON*, ad esempio il filo 11. Quindi il filo 10 deve essere *OFF*. Quindi l'interruttore C deve essere *off*  e l'interruttore D deve essere *ON*  .
- Il filo 6 è collegato a un componente  . Affinché l'uscita sia *OFF*, almeno uno dei fili in ingresso 12 e 13 deve essere *OFF*, quindi anche entrambi gli interruttori E e F possono essere *OFF*:  .
- Il filo 7 è collegato a un componente  . Affinché l'uscita sia *ON*, esattamente uno dei due fili in ingresso deve essere *ON*, ad esempio il filo 15. Quindi il filo 14 deve essere *OFF*. Quindi l'interruttore G deve essere *OFF*  e l'interruttore H deve essere *ON*  .

Esistono alternative con i componenti  , perché in questo caso è possibile decidere quale dei due fili in ingresso è *ON*. Inoltre, al componente  con il filo 6 come uscita si può decidere se

nessuno o uno dei due è *on*, perché in entrambi i casi l'uscita rimane *OFF*. Affinché l'uscita del componente  con il filo 6 sia *ON*, anche entrambi gli ingressi devono essere *ON*. In questo caso, i due ingressi del componente  con il filo 7 come uscita devono essere entrambi *ON* o entrambi *OFF*, in modo che il filo 7 sia *OFF*. In questo modo si ottengono 16 diverse combinazioni possibili:

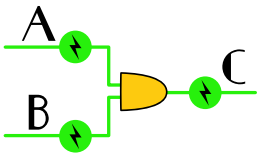
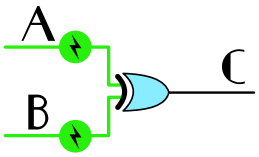
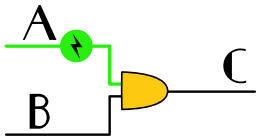
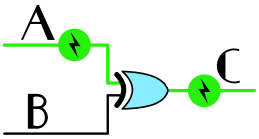
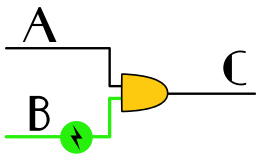
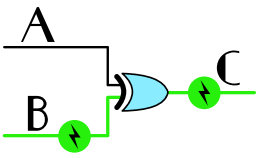
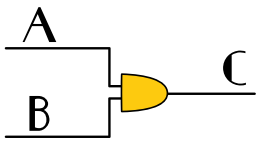
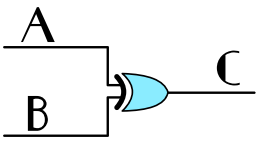
Interruttore								**Filo**	
A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**6**	**7**
sempre *ON*		**esattamente uno *ON***		**entrambi *ON*, se filo 6 è *ON*, altrimenti massimo uno *ON***		**esattament uno *ON*, se filo 7 è *ON*, altrimenti entrambi *ON* o *OFF***		**esattamente uno *ON***	
ON	ON	ON	OFF	ON	ON	ON	ON	ON	OFF
ON	ON	OFF	ON	ON	ON	ON	ON	ON	OFF
ON	ON	ON	OFF	ON	ON	OFF	OFF	ON	OFF
ON	ON	OFF	ON	ON	ON	OFF	OFF	ON	OFF
ON	ON	ON	OFF	ON	OFF	ON	OFF	OFF	ON
ON	ON	OFF	ON	ON	OFF	ON	OFF	OFF	ON
ON	ON	ON	OFF	ON	OFF	OFF	ON	OFF	ON
ON	ON	OFF	ON	ON	OFF	OFF	ON	OFF	ON
ON	ON	ON	OFF	OFF	ON	ON	OFF	OFF	ON
ON	ON	OFF	ON	OFF	ON	ON	OFF	OFF	ON
ON	ON	ON	OFF	OFF	ON	OFF	ON	OFF	ON
ON	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
ON	ON	ON	OFF	OFF	OFF	ON	OFF	OFF	ON
ON	ON	OFF	ON	OFF	OFF	ON	OFF	OFF	ON
ON	ON	ON	OFF	OFF	OFF	OFF	ON	OFF	ON
ON	ON	OFF	ON	OFF	OFF	OFF	ON	OFF	ON

Questa è l'informatica!

La corrente può passare o meno attraverso i fili di questo compito, quindi gli interruttori sono accesi o spenti. In informatica, tali stati rappresentano il valore di una *variabile booleana*. Questi sono spesso chiamati *vero* o *falso*, rispettivamente *1* o *0*.

I computer di oggi funzionano di solito solo con questi due stati. Uno dei motivi è che nel nucleo del computer sono incorporati miliardi di *transistor*, i cui ingressi e uscite sono anch'essi solo accesi o spenti.

Si possono quindi costruire *reti logici* a partire da diversi transistor. In questo compito sono presenti due reti di questo tipo: il componente  è una *porta AND* la cui uscita è attiva solo quando entrambi gli ingressi sono attivi. Il componente  è una *porta XOR* la cui uscita è attiva quando è attivo esattamente uno dei due ingressi. Si può anche scrivere questo come una *tabella della verità*:

Ingressi		Porta AND		Porta XOR	
Ingresso A	**Ingresso B**	**Immagine**	**Uscita C**	**Immagine**	**Uscita C**
ON	ON		ON		OFF
ON	OFF		OFF		ON
OFF	ON		OFF		ON
OFF	OFF		OFF		OFF

Altre porte comune sono la *porta OR*, la cui uscita è attiva quando almeno uno dei due ingressi è attivo, e l'*l'invertitore*, la cui uscita è attiva esattamente quando l'ingresso non è attivo. Spesso si utilizza una combinazione di una porta AND e di un invertitore, che può essere realizzata con un numero molto ridotto di transistor. Le tabelle di verità sono:

Ingresso A	Ingresso B	Uscita porta OR	Uscita invertitore
ON	ON	ON	OFF
ON	OFF	ON	ON
OFF	ON	ON	ON
OFF	OFF	OFF	ON

Ingresso	Uscita invertitore
ON	OFF
OFF	ON

Grazie ad abili combinazioni di *porte logiche*, un computer può eseguire calcoli complicati in modo molto rapido.

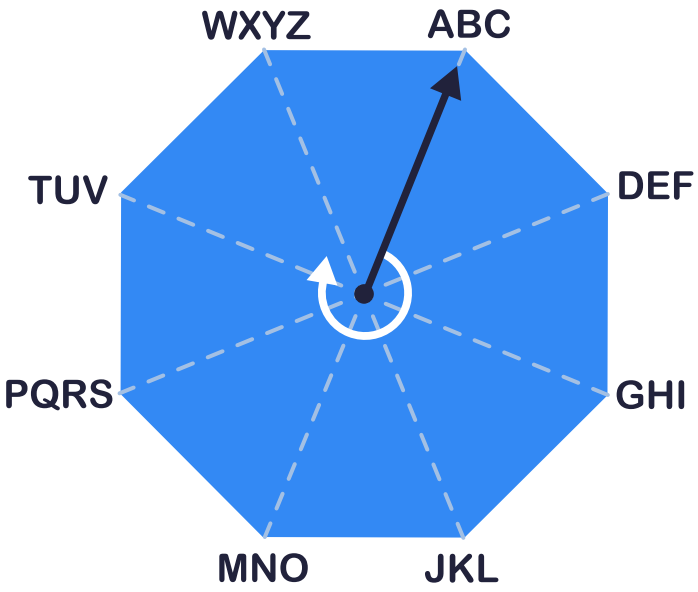
A un livello superiore, le porte logiche sono utilizzate anche nella programmazione: se l'esecuzione di una parte di un programma si basa su diverse condizioni, queste condizioni possono essere combinate con l'aiuto di operatori logici che funzionano esattamente nello stesso modo. Questo si riscontra anche nei programmi informatici. A volte un programma deve prendere «decisioni» su cosa fare dopo, a seconda che una cosa (o a volte diverse cose) sia già accaduta in precedenza.

Parole chiave e siti web

- Variabile booleana: `https://it.wikipedia.org/wiki/Variabile_booleana`
- Transistor: `https://it.wikipedia.org/wiki/Transistor`
- Rete logica: `https://it.wikipedia.org/wiki/Elettronica_digitale`
- Porta AND: `https://it.wikipedia.org/wiki/Porta_AND`
- Porta XOR: `https://it.wikipedia.org/wiki/Algebra_di_Boole#XOR`
- Tabella della verità: `https://it.wikipedia.org/wiki/Tabella_della_verità`
- Porta OR: `https://it.wikipedia.org/wiki/Porta_OR`
- Invertitore: `https://it.wikipedia.org/wiki/Invertitore`
- Porta logica: `https://it.wikipedia.org/wiki/Porta_logica`

8. Codice 8

Questo disco viene utilizzato per crittografare i testi in chiaro in testi cifrati:



All'inizio, il puntatore del disco è impostato su «ABC».

Ogni lettera viene crittografata singolarmente. A tal fine, vengono determinate due cifre:

- La prima cifra indica di quante posizioni è ruotato il puntatore in senso orario. Poi il puntatore viene posizionato sul blocco con la lettera da criptare.
- La seconda cifra indica la posizione della lettera da cifrare nel blocco puntato.

Ad esempio, la parola «RETE» è codificata come $53 - 42 - 51 - 32$.

Come si decifra il codice 52-12-43-54?

A) CASA

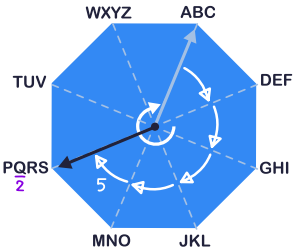
B) QUIZ

C) ROBOT

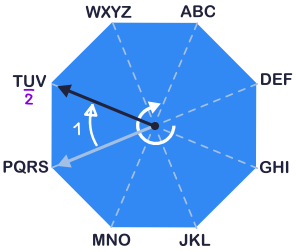
D) JAZZ

E) LUCE

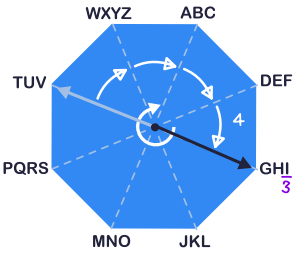
Soluzione



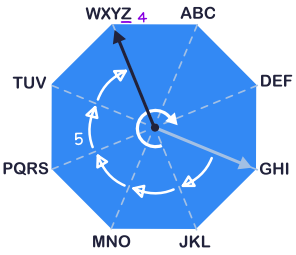
52 significa che il puntatore viene spostato dal blocco «ABC» al blocco «PQRS» (prima cifra 5) e che viene presa la seconda lettera «Q» (seconda cifra 2).



12 significa che il puntatore viene spostato dal blocco «PQRS» al blocco «TUV» (prima cifra 1) e che viene presa la seconda lettera «U» (seconda cifra 2).



43 significa che il puntatore viene spostato dal blocco «TUV» al blocco «GHI» (prima cifra 4) e che viene presa la terza lettera «I» (seconda cifra 3).



54 significa che il puntatore viene spostato dal blocco «GHI» al blocco «WXYZ» (prima cifra 5) e che viene presa la quarta lettera «Z» (seconda cifra 4).

Ciò significa che la risposta B) «QUIZ» è corretta.

Avresti potuto trovare questa soluzione più rapidamente: La risposta C) ROBOT non è possibile, perché è composta da cinque lettere, ma il testo cifrato ne rappresenta solo quattro. Poiché l'ultima lettera è codificata con un 4 come seconda cifra, può essere solo «S» o «Z». Solo le risposte B) e D) soddisfano questo requisito. La lettera che la precede deve provenire dal blocco di lettere di cinque giri in senso antiorario, cioè dal blocco «GHI». Ciò significa che la risposta può essere solo B) «QUIZ».

Questa è l'informatica!

Per migliaia di anni, l'uomo ha cercato di nascondere le informazioni in modo che solo i destinatari potessero decifrarle. Ciò che è iniziato con strisce di carta avvolte intorno a un bastone si è sviluppato attraverso cifrari a trasposizione come il «codice Cesare» e procedure di *crittografia polialfabetica* (come la «procedura Vigenére») fino alla moderna *crittografia a chiave pubblica* (come «GnuPG», che utilizza la «procedura RSA», tra le altre).

Il metodo di crittografia di questo compito è un metodo di crittografia polialfabetico, perché la stessa lettera non è necessariamente crittografata con lo stesso testo cifrato: la lettera «E» nell'esempio è crittografata come 42 all'inizio, ma come 32 alla fine. In linea di massima, tutti questi metodi di crittografia possono essere decifrati in modo semplice e veloce con l'aiuto dei computer.

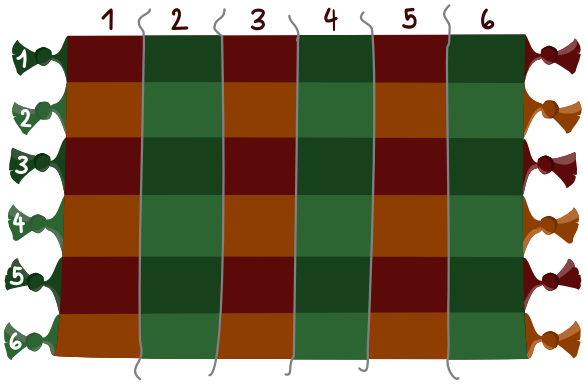
In questo caso, però, la decifrazione è semplicissima: esiste una sola chiave per criptare un testo. Anche se si potesse far partire la posizione iniziale del puntatore non dall'ABC ma da un qualche blocco, si avrebbero solo otto chiavi diverse... persino il codice Cesare, che ha più di 2000 anni, è «più sicuro». Ora si può ancora sostenere che il segreto non è la chiave ma il metodo di crittografia. Ma il *Principio di Kerckhoffs*, che Auguste Kerckhoffs (1835−1903) ha formulato nel 1883 e che è tuttora valido, chiarisce che la sicurezza di un *crittosistema* non deve basarsi sul mantenimento del segreto di un metodo di crittografia, perché questo potrebbe diventare troppo facilmente noto ad altri.

Parole chiave e siti web

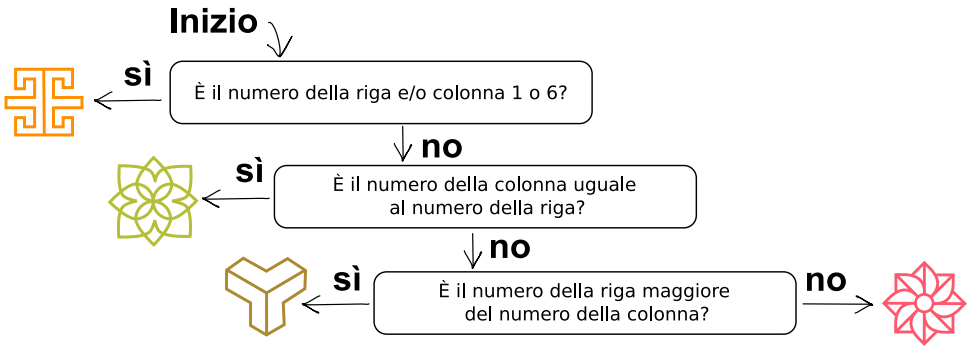
- Cifrario di Cesare: `https://it.wikipedia.org/wiki/Cifrario_di_Cesare`
- Cifrario polialfabetico: `https://it.wikipedia.org/wiki/Cifrario_polialfabetico`
- Cifrario: `https://it.wikipedia.org/wiki/Cifrario`
- Cifrario di Vigenère: `https://it.wikipedia.org/wiki/Cifrario_di_Vigenère`
- Crittografia asimmetrica: `https://it.wikipedia.org/wiki/Crittografia_asimmetrica`
- GNU Privacy Guard: `https://it.wikipedia.org/wiki/GNU_Privacy_Guard`
- RSA: `https://it.wikipedia.org/wiki/RSA_(crittografia)`
- Principio di Kerckhoffs: `https://it.wikipedia.org/wiki/Principio_di_Kerckhoffs`
- Crittosistema: `https://it.wikipedia.org/wiki/Crittosistema`
- Crittografia: `https://it.wikipedia.org/wiki/Crittografia`

9. Motivo del tappeto

Hale è un artista turco. Disegna un tappeto con una griglia di sei righe e sei colonne.

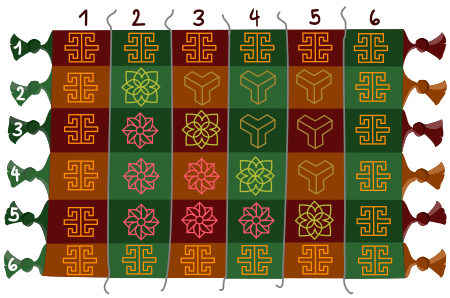


Hale numera le righe e le colonne. Quindi per ogni campo della griglia c'è il numero della riga e il numero della colonna. I commessi di Hale devono inserire un simbolo in ogni casella. Hale ha dato loro queste istruzioni per farlo:

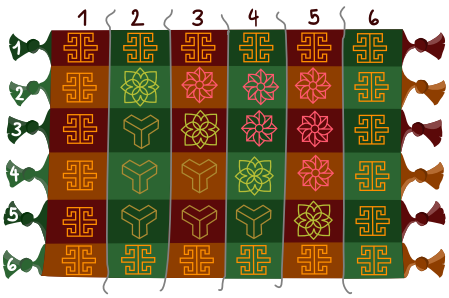


Come sarà il tappeto?

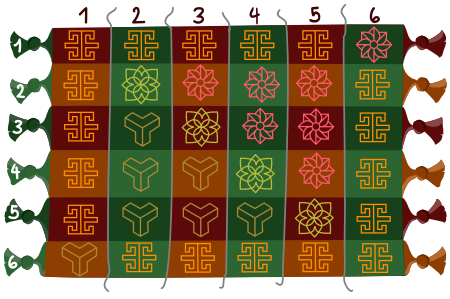
A)



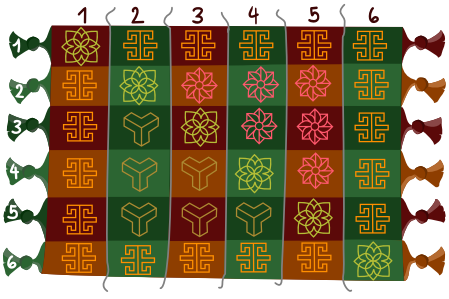
B)



C)



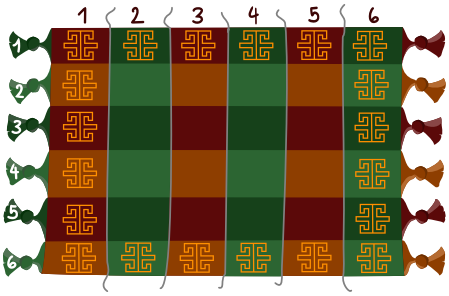
D)



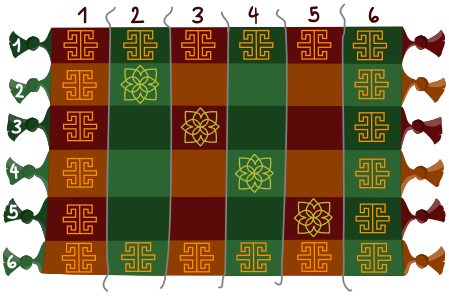
Soluzione

La risposta corretta è B).

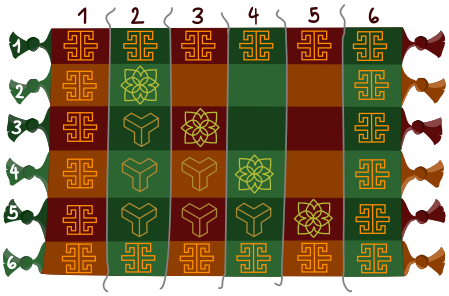
Alla prima domanda dell'immagine si risponde «sì» per tutti i quadrati sul bordo della griglia. Questo perché ogni campo del bordo si trova nella prima o nella sesta colonna o nella prima o nella sesta riga. A questi campi viene assegnato il simbolo  e si ottiene la seguente disposizione:



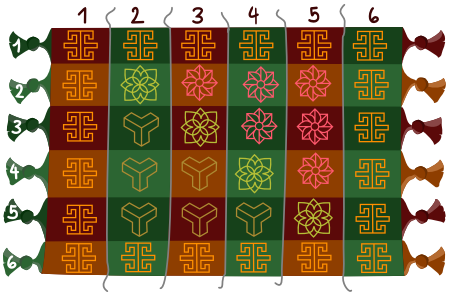
Alla seconda domanda si risponde con «sì» per tutti i campi sulla diagonale, perché sulla diagonale i numeri di colonna e di riga sono gli stessi. Questi campi ricevono il simbolo  e lo schema del tappeto si presenta come segue:



Secondo la terza domanda, tutti i campi il cui numero di riga è maggiore del numero di colonna ricevono il simbolo  .

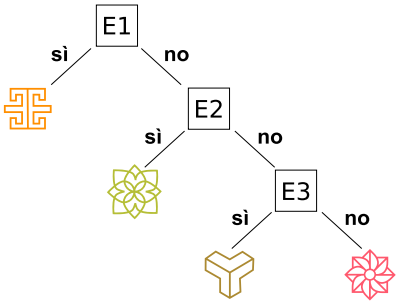


Per i campi rimanenti, alla terza domanda si risponde con «No». Ciò significa che il numero della riga non è maggiore del numero della colonna. Tutti questi campi sono riempiti con il simbolo  . In questo modo si ottiene il modello di tappeto della risposta B.



Questa è l'informatica!

L'immagine che l'artista Hale ha sviluppato come guida è chiamata *albero di decisione* in informatica. Come un vero e proprio albero, un albero di decisione è composto da rami. In ogni ramo (E1 - E3) c'è una domanda a cui si risponde con «Sì» o «No». Percorrendo l'albero da cima a fondo, rispondendo alle domande e seguendo le linee di corrispondenza, si arriva a una decisione.



Nel compito, l'albero di decisione è il fulcro delle istruzioni per tessere un tappeto. Ogni persona che utilizza queste istruzioni per la tessitura realizza esattamente lo stesso tappeto. In linea di principio, anche una macchina potrebbe produrre il tappeto, a patto che sia in grado di leggere e comprendere le istruzioni.

In informatica, un'istruzione unica di questo tipo è chiamata *algoritmo*. Se un algoritmo è scritto in un *linguaggio di programmazione* e può essere eseguito da un computer, si chiama *programma informatico*.

Nella vita di tutti i giorni, spesso si ha a che fare con programmi informatici che prendono decisioni: Il controllo del semaforo decide quando il semaforo pedonale diventa verde. Il sistema operativo del cellulare decide quando passare alla modalità di risparmio energetico. Il controllo automatico dei passaporti in aeroporto decide se il passaporto è valido.

Alla base di tutti questi programmi ci sono gli alberi di decisione.

Parole chiave e siti web

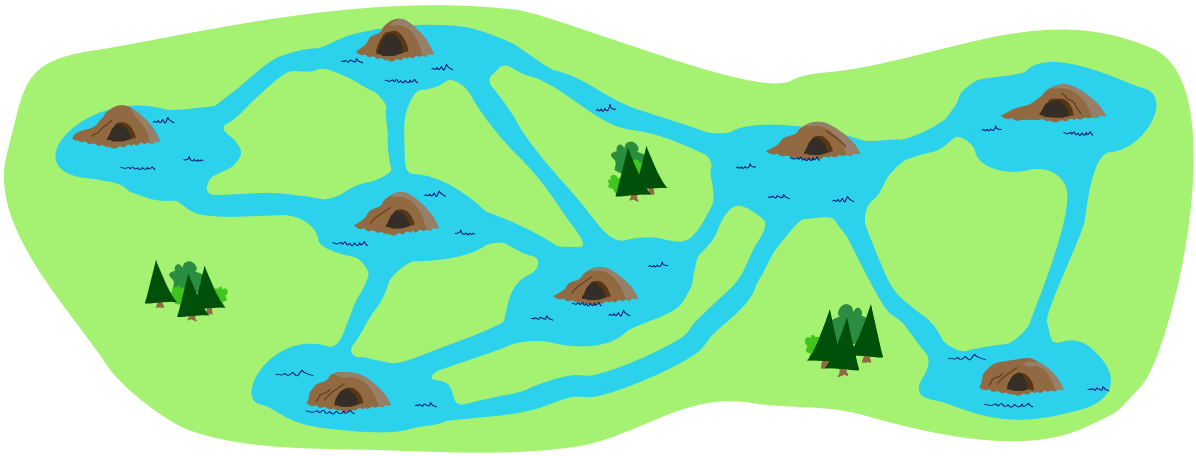
- Albero di decisione: `https://it.wikipedia.org/wiki/Albero_di_decisione`
- Algorithmo: `https://it.wikipedia.org/wiki/Algoritmo`
- Linguaggio di programmazione: `https://it.wikipedia.org/wiki/Linguaggio_di_programmazione`
- Programma: `https://it.wikipedia.org/wiki/Programma_(informatica)`

10. I vicini di Lili

Sulla mappa si possono vedere i castelli di otto castori. Due castori sono vicini di casa se un canale collega i loro castelli.

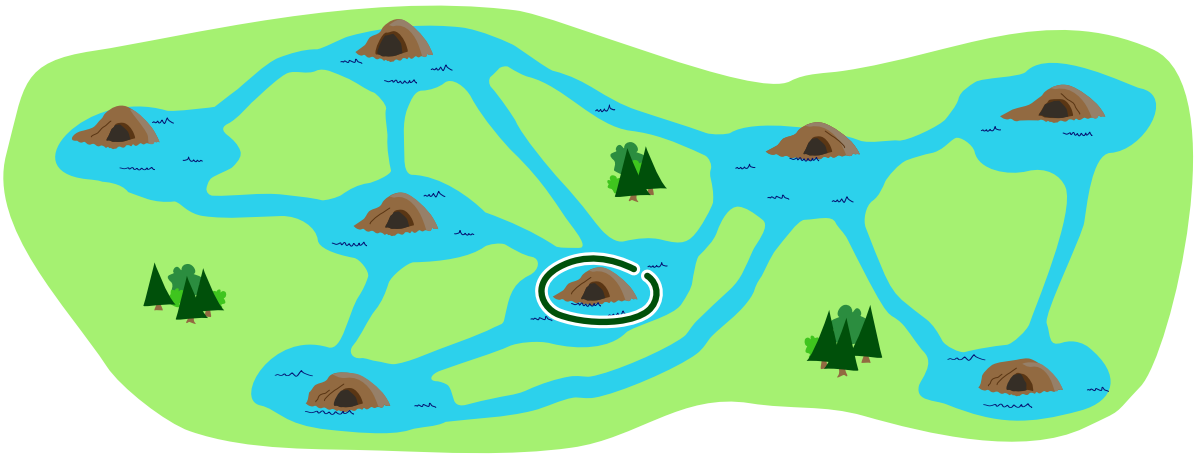
- Lili, Simon e Peter hanno ciascuno quattro vicini.
- Simon e Peter sono gli unici vicini di Nina.

In quale castello vive Lili?

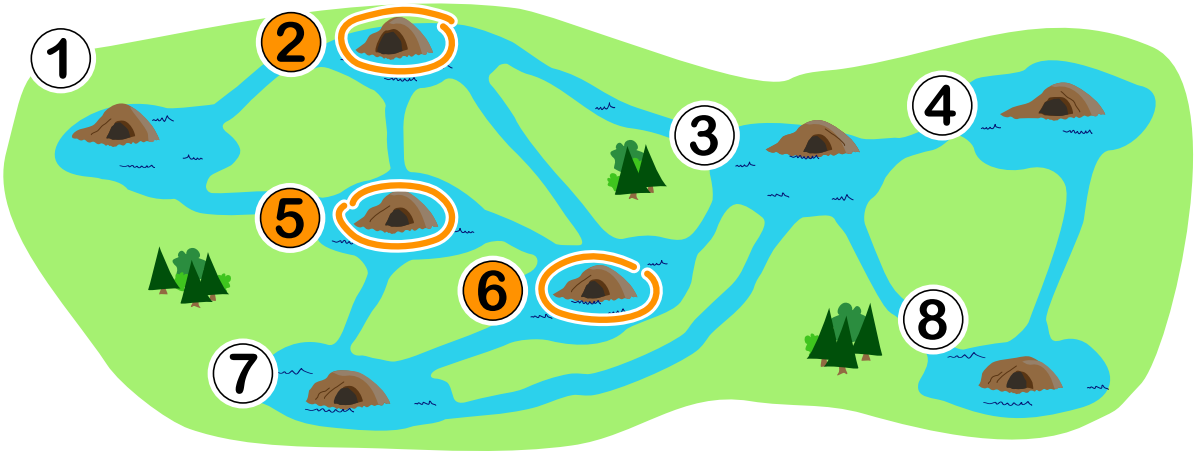


Soluzione

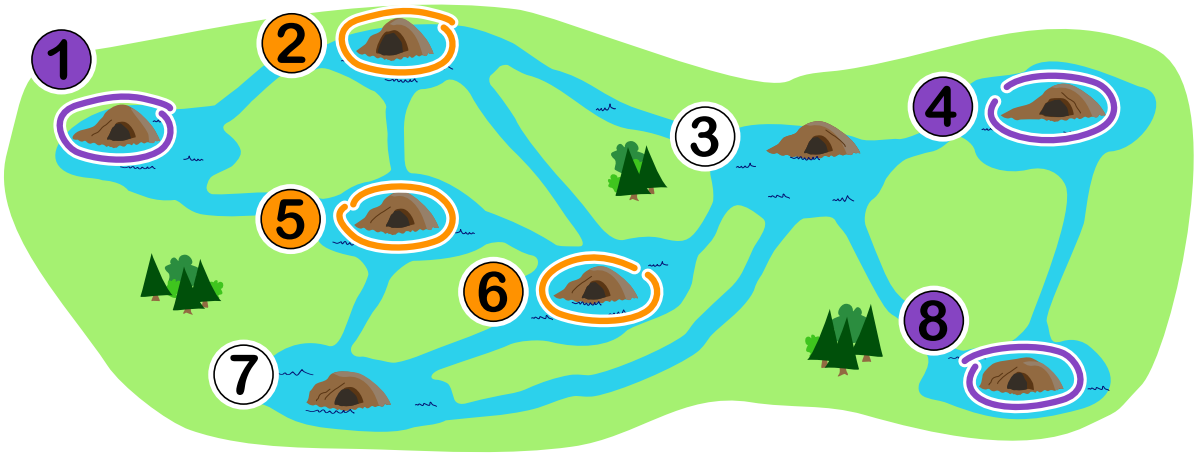
La soluzione corretta è:



Per risolvere il problema, è necessario concentrarsi sui canali tra i castelli. Dobbiamo identificare i castelli in cui vivono Lili, Peter o Simon. Poiché tutti hanno 4 vicini, ci devono essere esattamente quattro canali che partono da ciascuno dei loro castelli. Ci sono tre castelli di questo tipo: 2, 5 e 6.



Di conseguenza, Lili, Peter e Simon vivono ciascuno in uno di questi tre castelli. Ora dobbiamo scoprire in quale dei tre castelli vive Lili. Le altre due informazioni si riferiscono al castello di Nina. Da questi possiamo concludere che dal suo castello partono esattamente due canali. Quindi Nina vive in uno di questi castelli: 1, 4 o 8.



Siccome sappiamo che Simone e Pietro sono i due vicini di casa di Nina, possiamo concludere che

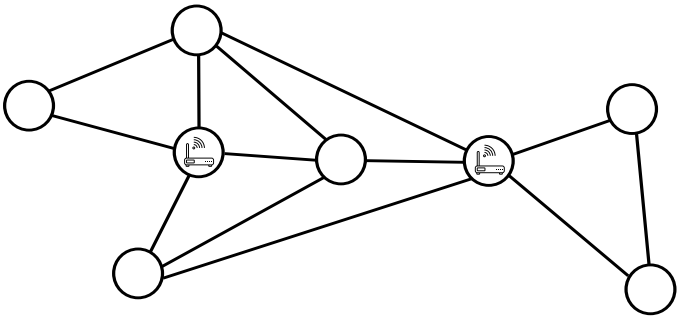
- Nina vive nel castello 1
- Simon e Peter vivono nei castelli 5 e 7 (o viceversa).

Quindi c'è solo un castello da cui partono quattro canali, che può essere il castello di Lili. È il castello 6!

Questa è l'informatica!

In questo compito, due castelli sono collegati da un canale. L'insieme dei castelli e dei canali forma una rete che mostra le relazioni tra tutti i castelli. Una tale rete di relazioni tra oggetti è chiamata *grafo* in informatica e matematica. Un grafo può essere considerato come un *insieme* di *vertici* collegati da *archi*. In questo compito, i castelli rappresentano i vertici e i canali gli archi.

Lo studio dei grafi è chiamato *teoria dei grafi*. Può essere utilizzato per modellare le relazioni a coppie tra gli oggetti. I grafi sono modelli matematici di strutture simili a reti in natura e in tecnologia. Ne sono un esempio le strutture sociali, le reti stradali, le reti informatiche, i circuiti elettrici, le reti di distribuzione o le molecole chimiche. I grafi possono essere utili per descrivere e risolvere i *problemi di rete*, come ad esempio trovare un buon posto per un router in un edificio o assicurarsi che ogni stanza di una casa abbia un segnale Wi-Fi forte.



Parole chiave e siti web

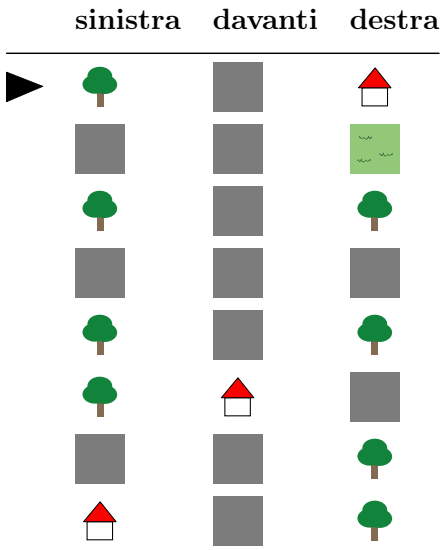
- Grafo: `https://it.wikipedia.org/wiki/Grafo`
- Insieme: `https://it.wikipedia.org/wiki/Insieme`
- Vertice: `https://it.wikipedia.org/wiki/Vertice_(teoria_dei_grafi)`
- Arco: `https://it.wikipedia.org/wiki/Glossario_di_teoria_dei_grafi#Arco`

11. La posta robotizzata

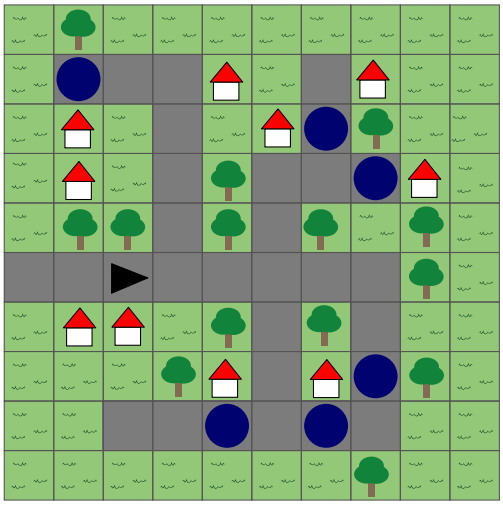
Il robot Tina consegna la posta. Tina utilizza una mappa suddivisa in campi. Tina si sposta lungo la strada verso una strada adiacente a sinistra, a destra o davanti (cioè non in diagonale).

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.

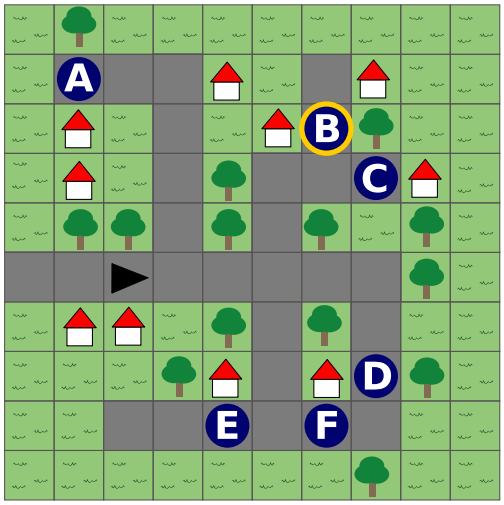


Quale dei punti blu scuro ● Tina raggiungerà alla fine del suo percorso?



Soluzione

La risposta corretta è il punto B.

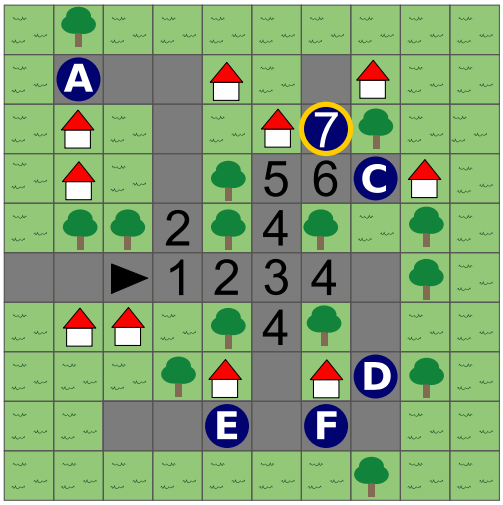


Passo	sinistra	davanti	destra
▶	🌳	■	🏠
1	■	■	
2	🌳	■	🌳
3	■	■	■
4	🌳	■	🌳
5	🌳	🏠	■
6	■	■	🌳
7	■	🌳	🌳

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.

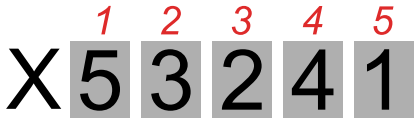
Le auto a guida autonoma sono esempi famosi di questi robot. Sono dotati di numerosi sensori che non solo misurano la velocità o la posizione corrente, ma anche la distanza dal ciglio della strada e rilevano gli oggetti presenti sulla strada o a bordo strada e molto altro ancora. Queste informazioni 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`

12. Sequenze di dati

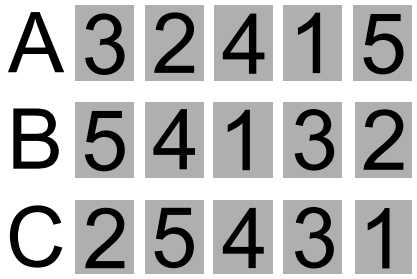
Qui possiamo vedere una sequenza di numeri di nome X. Nelle posizioni da 1 a 5 della sequenza X si trovano i seguenti numeri: 5, 3, 2, 4, 1.



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.



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

A) 1

B) 2

C) 3

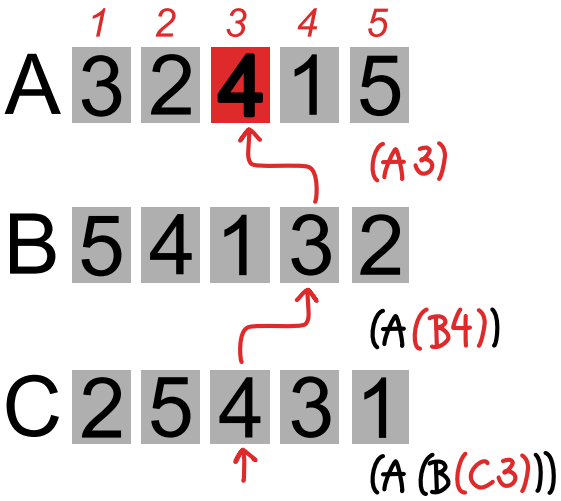
D) 4

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.

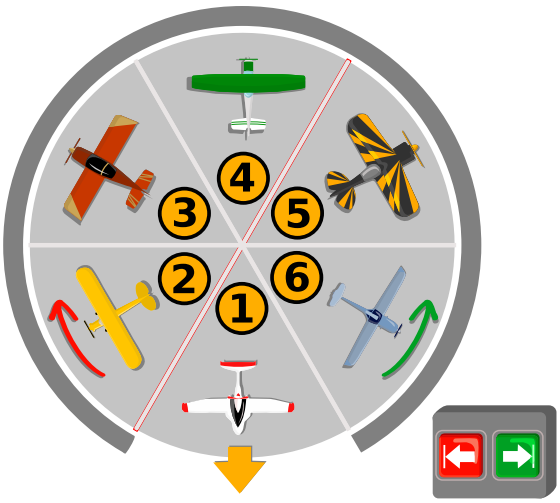
Statico o dinamico: se una struttura di dati di una sequenza contiene numeri, questi numeri possono anche indicare posizioni nella stessa o in un'altra sequenza. Questo viene spesso utilizzato in informatica per il cosiddetto indirizzamento indiretto: L'indirizzo o la posizione in una sequenza non è 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`
- 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`

13. Capannone rotante

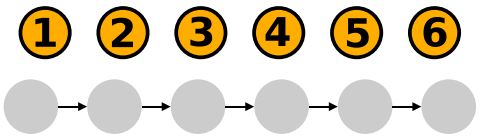
Nel campo di aviazione di Beavertown, sei aerei sono parcheggiati in un capannone. Sono su una piattaforma rotante, parcheggiati in sei posizioni diverse. All'esterno sono presenti due tasti freccia ⬅ ➡. Con un solo tasto è possibile ruotare l'unità di rotazione esattamente di una posizione di parcheggio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

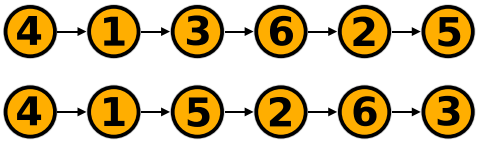
Ma qual è il caso peggiore? In quale ordine devono essere premuti più spesso i tasti?

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`
- Timsort: `https://it.wikipedia.org/wiki/Timsort`
- O grande: `https://it.wikipedia.org/wiki/O-grande`

14. 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.

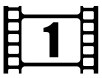
Ora Ada vuole convincere il minor numero possibile di amici a cambiare la propria valutazione, in modo che alla fine ci sia un preferito.

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
	4: Nancy, Niklaus, Grace e Rozsa
	3: Niklaus, Edsger e Rozsa
	3: Niklaus, Edsger e Rozsa
	3: Nancy, Edsger e Rozsa
	3: Nancy, Grace e Edsger
	2: Niklaus e Rozsa
	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, Rzosa 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

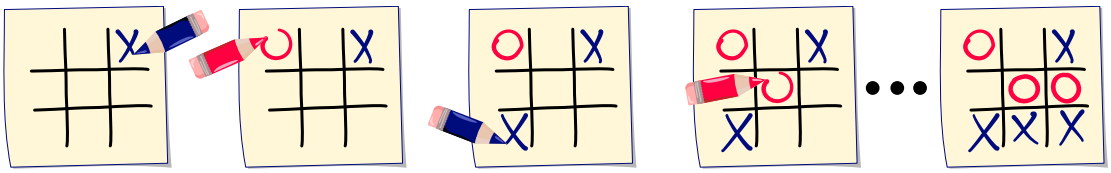
- Algoritmo: `https://it.wikipedia.org/wiki/Algoritmo`

15. Tris

Il tris è un gioco per due persone.

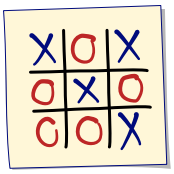
In una griglia con 3×3 caselle, i due giocatori riempiono a turno un simbolo in una casella vuota: un giocatore X, l'altro O. Il primo giocatore che riempie tre caselle in fila, in colonna o in diagonale con il proprio simbolo vince e la partita è finita. Se tutte le caselle sono riempite e nessuno ha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

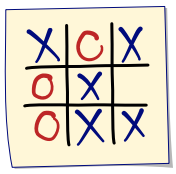
Qui si possono vedere i punteggi di una possibile partita: le prime 4 mosse e l'ultima mossa. Il giocatore con X vince.

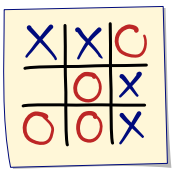


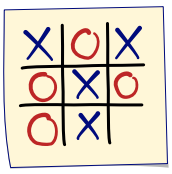
Il punteggio alla fine di una partita è chiamato punteggio finale. Le regole del gioco specificano esattamente come possono essere compilati i campi con X e O e quando il gioco termina.

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

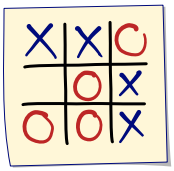
A) 

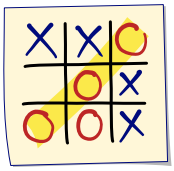
B) 

C) 

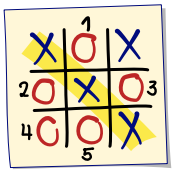
D) 

Soluzione

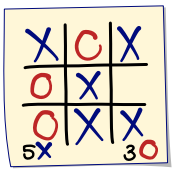
La risposta C è corretta: 



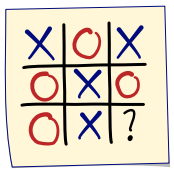
La risposta C è corretta perché un giocatore aveva vinto (tre O in una diagonale) e poi non sono state fatte altre mosse.



La risposta A non è corretta. Il giocatore X ha vinto la partita, ma il giocatore O 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 X ma solo 3 campi con O. Questo non è possibile, perché il numero di caratteri O- e il numero di caratteri X- 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 X e il numero di O 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.

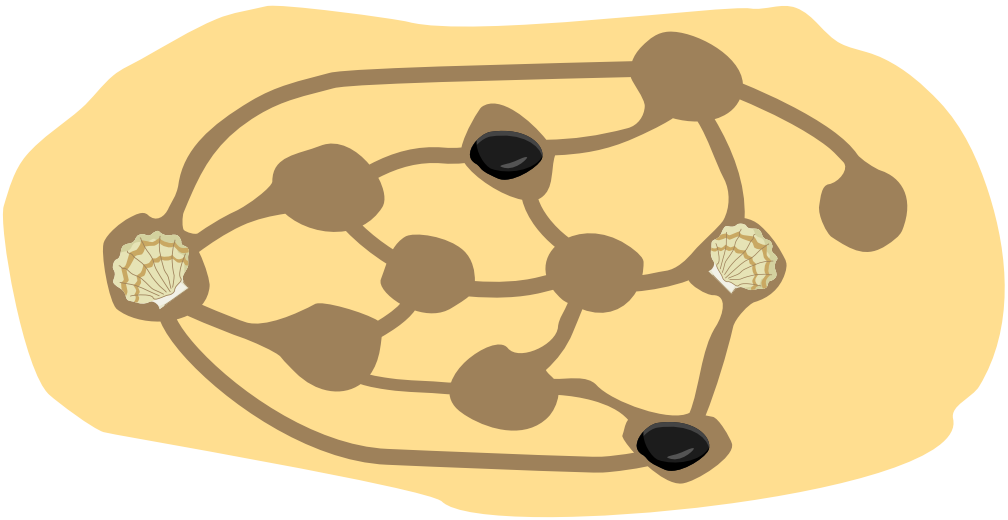
Parole chiave e siti web

- Tris: `https://it.wikipedia.org/wiki/Tris_(gioco)`
- Interprete: `https://it.wikipedia.org/wiki/Interprete_(informatica)`
- Linguaggio di programmazione:
  `https://it.wikipedia.org/wiki/Linguaggio_di_programmazione`

16. 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.

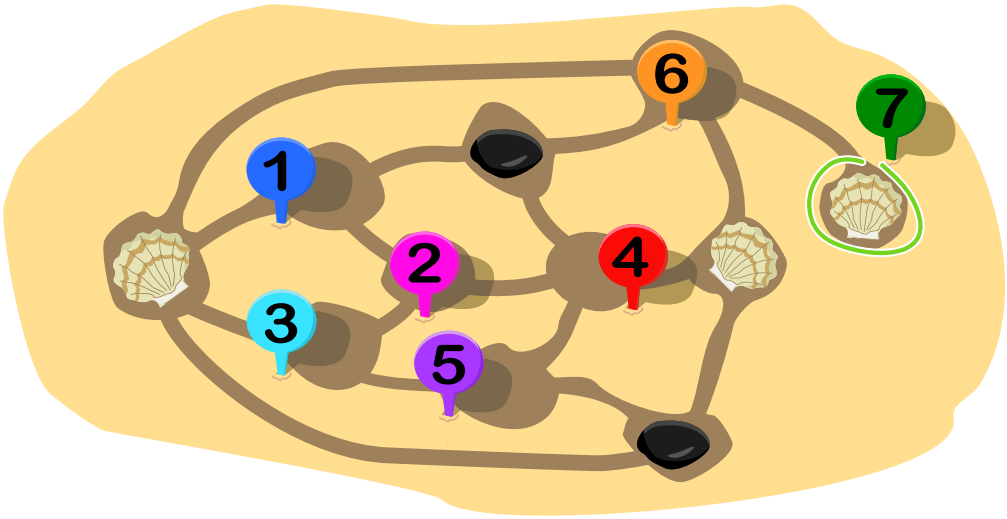
A turno, i giocatori posizionano uno delle loro pedine in uno spazio libero. Il primo giocatore che posiziona due proprie pedine in due cavità direttamente collegate perde. L'immagine mostra il punteggio dopo alcune mosse.



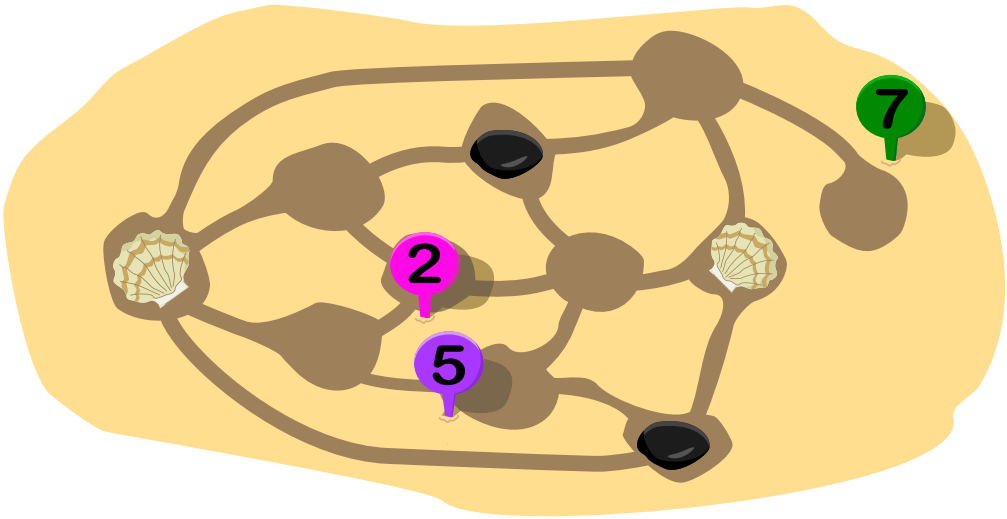
È il turno di Ann. In quale delle cavità libere deve posizionare la sua prossima conchiglia per assicurarsi la vittoria?

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.

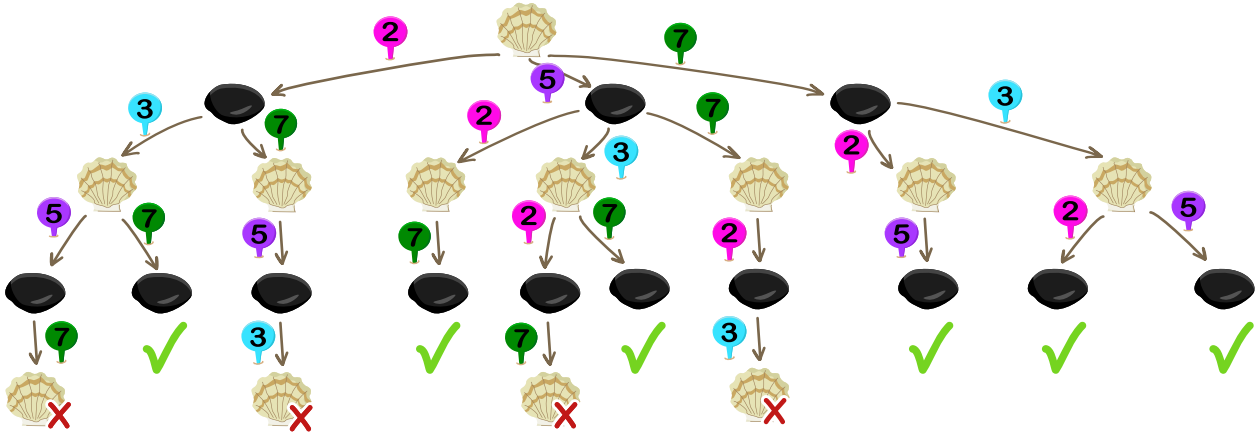
Se Ann giocasse 2 al punteggio della figura, Bob potrebbe giocare 7 al prossimo colpo. Dopo di che, Ann avrebbe dovuto giocare 5, Bob avrebbe dovuto giocare 3 e Ann avrebbe perso.

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

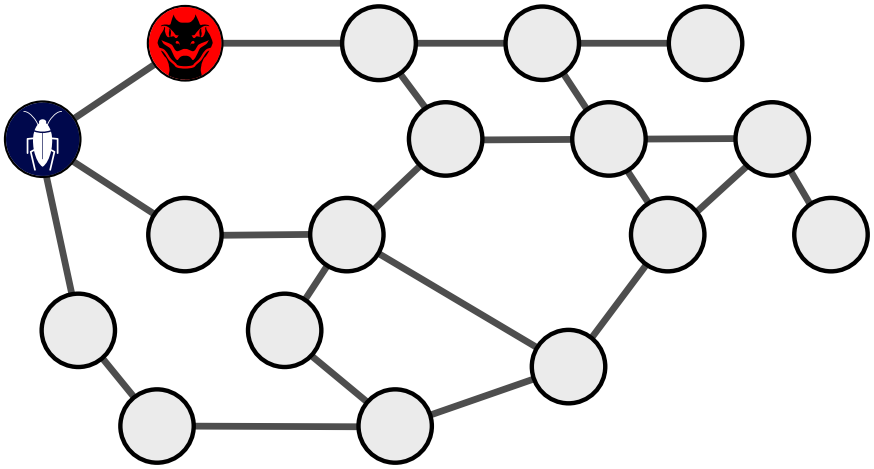
- Teoria dei giochi: `https://it.wikipedia.org/wiki/Teoria_dei_giochi`
- John Horton Conway: `https://it.wikipedia.org/wiki/John_Conway`

17. 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.

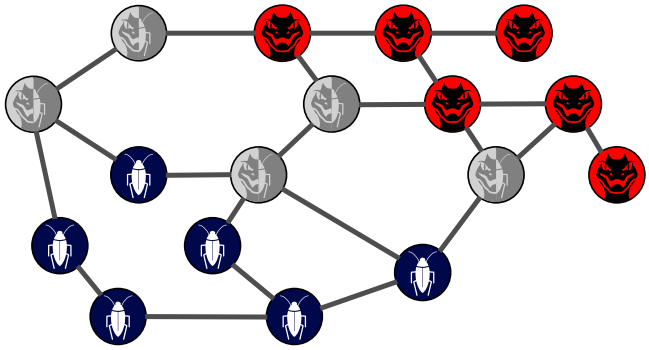
Qui sotto puoi vedere la rete di computer con i nodi e le loro connessioni dirette. I due nodi infettati all'inizio sono contrassegnati. Dopo qualche giorno, tutti i nodi vengono infettati da un virus o addirittura spenti.

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

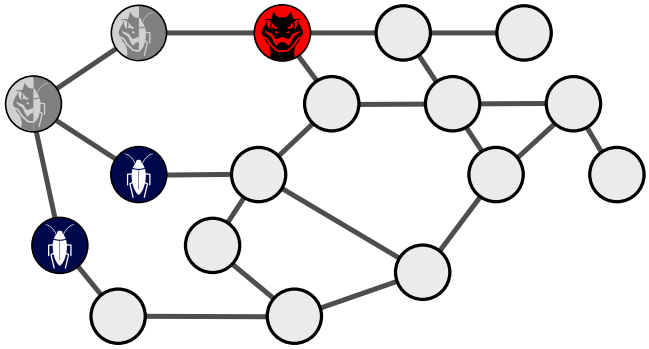


Soluzione

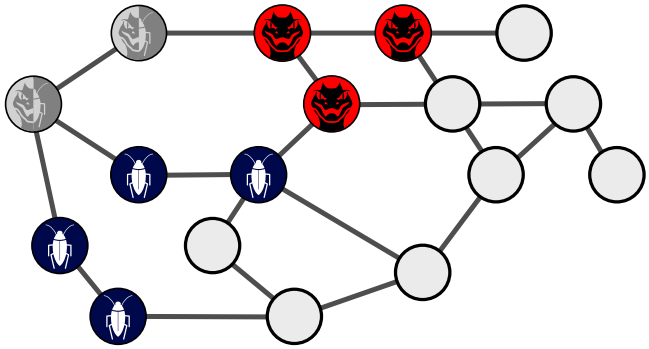
Dopo 5 giorni, tutti i nodi della rete vengono infettati o spenti. Questa è la soluzione corretta:



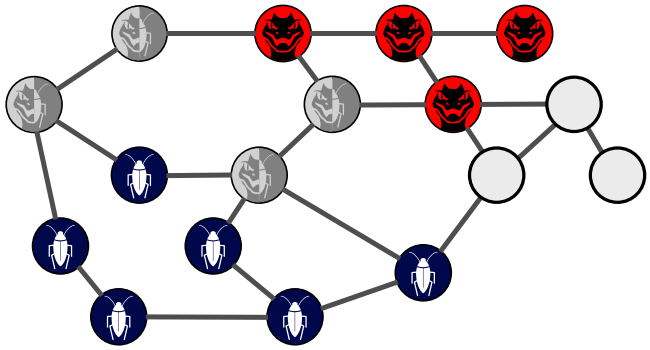
Dopo 1 giorno, cinque nodi della rete sono stati infettati. I due nodi infettati all'inizio sono ora infettati da entrambi i virus e quindi spenti:



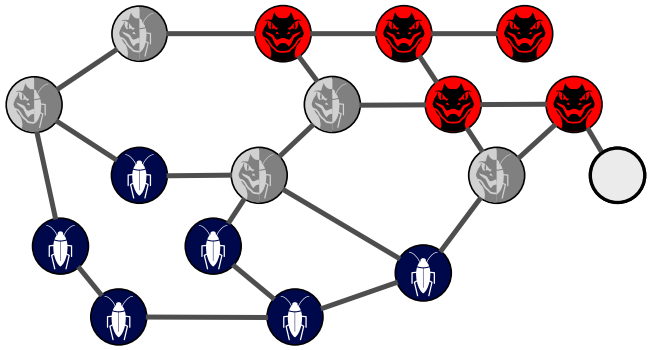
Dopo 2 giorni, vengono infettati altri quattro nodi:



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`
- Virus: `https://it.wikipedia.org/wiki/Virus_(informatica)`
- Botnet: `https://it.wikipedia.org/wiki/Botnet`
- Guerra cibernetica: `https://it.wikipedia.org/wiki/Guerra_cibernetica`

A. Autori dei quesiti

Gulgun Afacan
Esraa Almajhad
Waël Almoman
Leo Barichello
Liam Baumann
Wilfried Baumann
Linda Björk Bergsveinsdóttir
Daniela Bezáková
Marta J. Burzanska
Sarah Chan
Byeonggyu Cho
Darija Dasović
Christian Datzko
Susanne Datzko
Nora A. Escherle
Gerald Futschek
Christian Giang
Adam Grodeck
Yasemin Gülbahar
Benjamin Hirsch
Alisher Ikramov
Thomas Ioannou
Hakin Kim
Jihye Kim
Seulki Kim
Vaidotas Kinčius
Víctor Koleszar
Lidija Kralj
Regula Lacher
Taina Lehtimäki
Marielle Léonard
Inggriani Liem
Karolína Miková
Zoran Milevski
Madhavan Mukund
Ágnes Erdősné Németh
Ilze Nilandere
Mārtiņš Opmanis
Jean-Philippe Pellet
Margot Phillipps
Zsuzsa Pluhár
Wolfgang Pohl
Susannah Quidilla
Lorenzo Repetto
Kirsten Schlüter
Giovanni Serafini
Yeh Yi Shan
Bernadette Spieler
Alieke Stijf
Goran Sukovic
Monika Tomcsányiová
Ahto Truu

 Jiří Vaníček

 Troy Vasiga

 Michael Weigend

 Kyra Willekes

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

UBS

http://www.ubs.com/

verkehrshaus.ch

http://www.verkehrshaus.ch/
Musée des transports, Lucerne

i-factory (Musée des transports, Lucerne)

senarclens
leu+partner
strategische kommunikation

http://senarclens.com/
Senarclens Leu & Partner

ABZ
AUSBILDUNGS- UND BERATUNGSZENTRUM
FÜR INFORMATIKUNTERRICHT

http://www.abz.inf.ethz.ch/
Ausbildungs- und Beratungszentrum für Informatikunterricht der ETH Zürich.

hep/ haute école pédagogique vaud

http://www.hepl.ch/
Haute école pédagogique du canton de Vaud

Scuola universitaria professionale
della Svizzera italiana
SUPSI

http://www.supsi.ch/home/supsi.html
La Scuola universitaria professionale della Svizzera italiana (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 CoetryLab 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.

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.