INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Quesiti e soluzioni 2020

5º e 6º anno scolastico

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

A cura di:

Lucio Negrini, Christian Giang, Susanne Datzko, Fabian Frei,
Juraj Hromkovič, Regula Lacher, Jean-Philippe Pellet

0101001101010110010010010000001001011010101001101010010100100101000010100101

SSII

www.svia-ssie-ssii.ch
schweizerischerverein für informatikinderausbildung//société suisse pour l'informatique dans l'enseignement//società svizzera per l'informatica nell'insegnamento

Hanno collaborato al Castoro Informatico 2020

Susanne Datzko, Fabian Frei, Martin Guggisberg, Lucio Negrini, Gabriel Parriaux, Jean-Philippe Pellet

Capo progetto: Nora A. Escherle

Un particolare ringraziamento per il lavoro sui quesiti del concorso Svizzero va a:
Juraj Hromkovič, Michael Barot, Christian Datzko, Jens Gallenbacher, Dennis Komm, Regula Lacher, Peter Rossmanith: ETH Zürich, Ausbildunges- und Beratungszentrum für Informatikunterricht

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: Bebras.org
Wolfgang Pohl, Hannes Endreß, Ulrich Kiesmüller, Kirsten Schlüter, Michael Weigend: Bundesweite Informatikwettbewerbe (BWINF), Germania
Wilfried Baumann, Anoki Eischer: 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, Arne Heijenga, Dave Oostendorp, Andrea Schrijvers, Alieke Stijf, Kyra Willekes: cuttle.org, Olanda
Chris Roffey: University of Oxford, Regno Unito

Per il supporto durante le settimane del concorso ringraziamo:
Hanspeter Erni: Direttore scuola media di Rickenbach
Gabriel Thullen: Collège des Colombières
Beat Trachsler: Scuola cantonale di Kreuzlingen

Christoph Frei: Chragokyberneticks (Logo Informatik-Biber Schweiz)
Dr. Andrea Leu, Maggie Winter, Brigitte Manz-Brunner: 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 2020 è stato organizzato dalla Società Svizzera per l'Informatica nell'Insegnamento SSII con il sostegno della fondazione Hasler.

HASLERSTIFTUNG

Questo quaderno è stato creato il 9 settembre 2021 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'01.12.2020.

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 nell'ambito del programma di promozione «FIT in IT».

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 2020 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

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 ha iniziato 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.

Per ulteriori informazioni:

SVIA-SSIE-SSII Società Svizzera per l'Informatica nell'Insegnamento
Castoro Informatico
Lucio Negrini

`https://www.castoro-informatico.ch/it/kontaktieren/`
`https://www.castoro-informatico.ch/`

Indice

1. Spettacolo teatrale

Uno spettacolo teatrale presenta una saggia principessa, un nobile cavaliere, un re bello e un drago malvagio. All'inizio il palco è vuoto. Durante lo spettacolo queste quattro figure entrano ed escono dal palco nel seguente ordine:

Primo atto		**Atto secondo**
Il re entra in scena		Il drago entra in scena
La principessa entra in scena		Il cavaliere entra in scena
Il re esce di scena	PAUSA	Il drago esce di scena
Il drago entra in scena		La principessa entra in scena
La principessa esce di scena		Il cavaliere esce di scena
Il drago esce di scena		La principessa esce di scena
Pausa		**Fine**

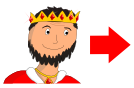
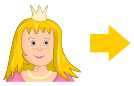
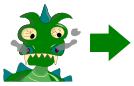
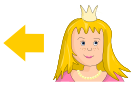
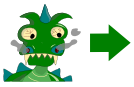
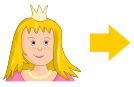
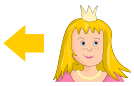
Cosa non succederà?

A) La principessa e il cavaliere sono sul palco insieme.
B) Il re e il drago sono sul palco insieme.
C) Il cavaliere entra in scena dopo la pausa.
D) Il cavaliere e il drago sono sul palco insieme.

Soluzione

La risposta corretta è B) «Il re e il drago sono sul palco insieme», perché questa affermazione non è mai vera durante tutto lo spettacolo.

Si può analizzare passo per passo:

Azione	Re sul palco?	Principessa sul palco?	Drago sul palco?	Cavaliere sul palco?	Risposte corrispondenti
Atto primo					
	Sì	No	No	No	
	Sì	Sì	No	No	
	No	Sì	No	No	
	No	Sì	Sì	No	
	No	No	Sì	No	
	No	No	No	No	
Pausa					
Atto secondo					
	No	No	Sì	No	
	No	No	Sì	Sì	C), D)
	No	No	No	Sì	
	No	Sì	No	Sì	A)
	No	Sì	No	No	
	No	No	No	No	
Fine					

Per ogni risposta è possibile verificare se l'affermazione in essa contenuta è vera o meno, esaminando le righe delle tabelle.

Nella risposta A), si cerca una riga in cui sia la principessa che il cavaliere sono sul palco insieme. Questo è il caso della quarta riga del secondo atto, perché la principessa entra nel palco dove il cavaliere si trova dalla seconda riga e rimane fino alla quinta riga. L'affermazione della risposta A) è quindi corretta almeno in un certo momento.

Nella risposta D), si cerca una riga in cui il cavaliere e il drago sono sul palco insieme. Questo è il caso della seconda riga del secondo atto, perché il cavaliere entra sul palco nella seconda riga, mentre il drago è già entrato sul palco nella prima riga e rimane fino alla terza riga. L'affermazione della risposta D) è quindi corretta almeno in un certo momento.

Nella risposta C), l'affermazione è di natura diversa. Se questo è vero, il cavaliere non deve essere entrato in scena durante tutto il primo atto. Qui bisogna guardare la colonna del cavaliere per il primo atto. C'è scritto «No» ovunque, quindi il cavaliere in realtà non è entrato in scena durante tutto il primo atto. Ma poi entra nella seconda riga del secondo atto, quindi anche l'affermazione della risposta C) è vera.

Se l'affermazione della risposta B) fosse vera, il re e il drago dovrebbero stare insieme sul palco in qualche riga. Ma in nessuna delle dodici righe c'è un «Sì» in entrambe le colonne. Il re esce di scena già nella terza riga del primo atto e non entra più in scena fino alla fine. Il drago, invece, non entra in scena fino alla quarta riga del primo atto. Forse i due si incontrano dietro il palco, ma sul palco non sono mai insieme. Pertanto l'affermazione della risposta B) non è corretta. Quindi B) è la risposta corretta.

Questa è l'informatica!

Anche se si può immaginare vividamente una storia basata sul corso dello spettacolo, solo una caratteristica è importante per ogni personaggio in questo compito: è o non è sul palco in un certo momento? Questa limitazione della visione a certe caratteristiche è chiamata *astrazione*.

Nell' informatica, tali astrazioni possono essere formulate molto bene. Per ognuna delle quattro figure si definisce una cosiddetta *variabile*, che risponde alla domanda se la figura è attualmente in scena. Le quattro variabili sono: «Re sul palco?», «Principessa sul palco?», «Drago sul palco?» e «Cavaliere sul palco?». Durante lo spettacolo, le risposte a queste domande cambiano continuamente; per ogni domanda la risposta è a volte «sì» e a volte «no». Nell' informatica, chiamiamo la risposta attuale ad una domanda il *valore* attuale della variabile associata. Il valore di una variabile può quindi cambiare continuamente (in matematica è diverso, le variabili non cambiano i loro valori nel tempo). La tabella nella spiegazione della risposta mostra le quattro variabili e i valori corrispondenti in qualsiasi momento.

Esiste anche un altro modo di considerare lo spettacolo. In ogni momento guardiamo quali personaggi sono sul palco in quel momento (quindi guardiamo i valori attuali delle quattro variabili.) Ogni possibile combinazione di figure la chiamiamo uno *stato* del palco. Così, quando una figura entra o esce dal palco, lo stato del palco cambia. Possiamo chiamarla quindi una *transizione* del palco da uno stato all'altro. Se si prende un pezzo di carta e si disegna un cerchio separato per ogni possibile stato (cioè ogni combinazione di figure), si può considerare il tutto come un'astrazione del palco.

Inoltre, è possibile disegnare le possibili transizioni come frecce che portano da uno stato all'altro. Se facciamo anche questo, abbiamo quello che in informatica chiamiamo *diagramma di stato* del palco.

All'inizio dello spettacolo il palco è vuoto. Per questo motivo chiamiamo lo stato corrispondente *stato iniziale*. Ora possiamo disegnare il percorso del gioco come un percorso nel diagramma di stato. Il percorso inizia nello stato iniziale e poi segue le frecce che corrispondono all'azione.

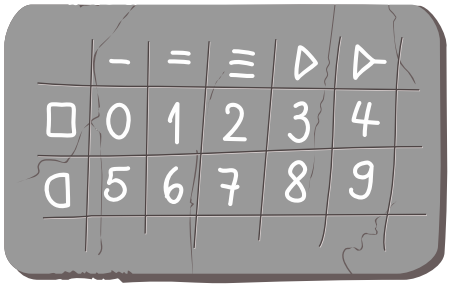
I diagrammi di stato sono molto importanti nell'informatica. Con quasi tutti i sistemi complicati, ad un certo punto bisogna pensare al diagramma di stato. Ma per gli esseri umani è spesso molto noioso lavorare con questi stati e transizioni astratti. I computer, invece, sono estremamente bravi per questo. Pertanto, vale la pena se le persone possono rappresentare i loro problemi con i diagrammi di stato in modo tale che i computer possano poi risolverli.

Parole chiave e siti web

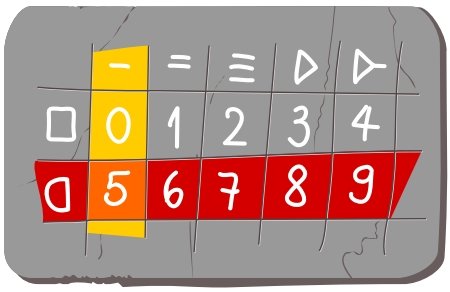
- Variabili: `https://it.wikipedia.org/wiki/Variabile_(informatica)`
- Diagramma di stato: `https://it.wikipedia.org/wiki/Diagramma_di_stato_(informatica)`

2. Anno di costruzione del castello

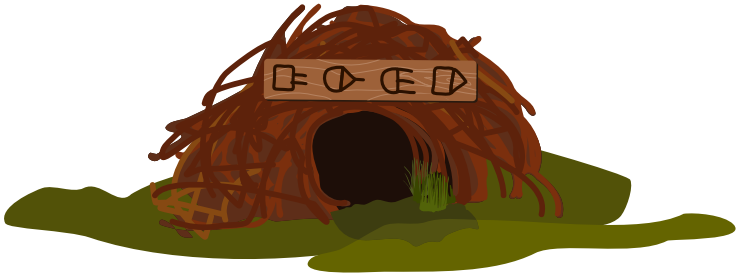
Sul cartello sopra l'ingresso di ogni castello dei castori è indicato l'anno di costruzione. I castori usano i loro caratteri per i numeri. La seguente tabella mostra come le cifre possono essere utilizzate per comporre i caratteri dei castori:



Ad esempio, i castori utilizzano la cifra «5» per formare il nuovo carattere  , che è assemblato nel seguente modo:



Questo è il castello di Cleveria:

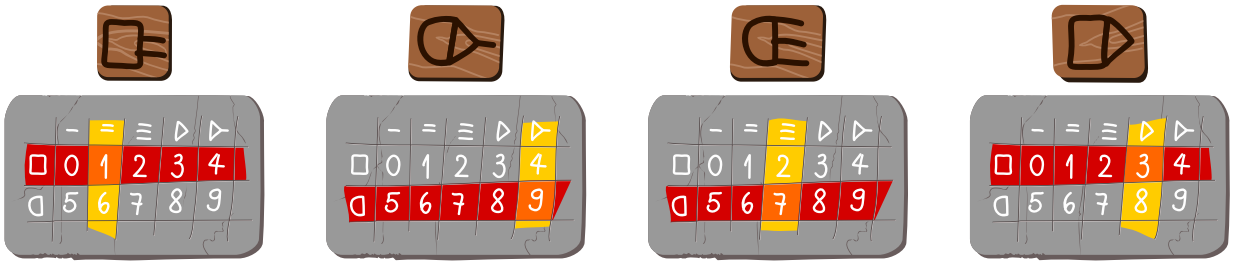


In quale anno è stato costruito il castello di Cleveria?

A) 0978
B) 1574
C) 1923
D) 1973
E) 1993
F) 2973
G) 6378

Soluzione

È possibile scoprire l'anno di costruzione del castello scegliendo la riga e la colonna appropriata per ogni simbolo. All'intersezione tra la colonna e la riga troverete il numero che cercate.



Le quattro cifre nell'ordine corretto danno il numero 1973.

Questa è l'informatica!

Mantenere la segretezza delle informazioni e proteggere i dati è un compito che risale a 4000 anni fa. A questo scopo sono stati sviluppati e utilizzati innumerevoli linguaggi segreti. Oggi la sicurezza dei dati è uno dei temi centrali dell'informatica. Uno dei metodi per proteggere i dati da letture non autorizzate è la *crittografia*. La cifratura trasforma un testo in chiaro in un testo cifrato. Ricostruire il testo in chiaro dal *testo cifrato* si chiama *decifrare*. La scienza del testo cifrato si chiama *crittologia*.

Le culture antiche utilizzavano per lo più scritture segrete, che venivano create codificando le lettere con altre lettere o con caratteri completamente nuovi. Il cifrario seguente è stato sviluppato specialmente per la competizione del castoro informatico, ma si basa su un concetto dell'antica Palestina. La regola di sicurezza dell'epoca era che si usavano solo codici segreti che si potevano imparare facilmente a memoria. Mantenere una descrizione scritta del codice segreto era considerato un rischio troppo grande. Una tabella, come si usa qui, è facile da imparare a memoria. Il famoso codice segreto dei massoni si basa su questo principio.

Invece di limitarsi a mettere insieme nuovi caratteri per le cifre, si possono anche inventare nuovi codici segreti per i testi. Per fare questo, si scrivono tutte le lettere in una tabella e si inventano nuovi simboli per le colonne e le righe, ottenendo così nuovi caratteri per tutte le lettere.

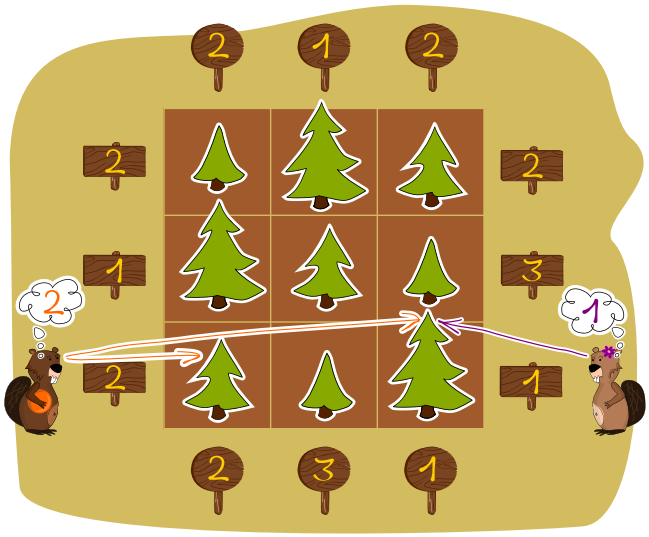
Parole chiave e siti web

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

3. 3×3 sudoku con gli alberi

I castori piantano abeti in fila. Gli abeti hanno tre diverse altezze (1 , 2 e 3) e in ogni fila c'è esattamente un abete di ogni altezza.

Quando i castori guardano una fila di abeti da un'estremità, **non** possono vedere gli abeti più bassi nascosti dietro gli abeti più alti.



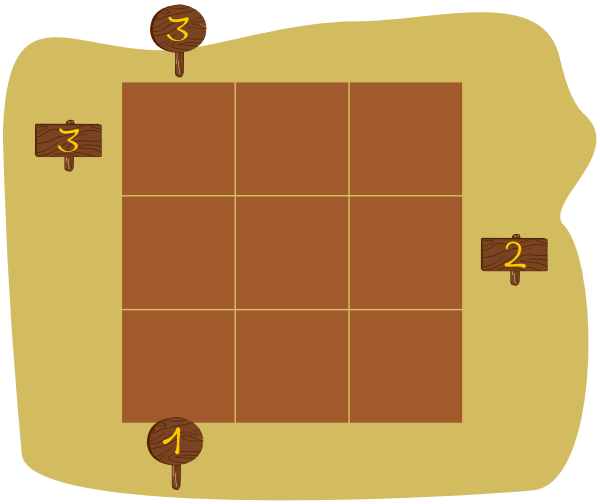
Alla fine di ogni fila di abeti c'è un cartello che indica quanti abeti un castoro può vedere da quel punto.

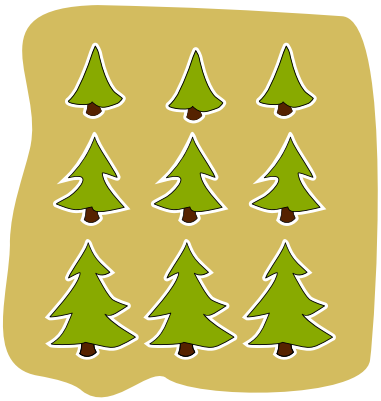
Ora i castori piantano nove abeti in un campo 3×3, come nell'esempio a destra.

Si applicano le seguenti regole:

- in ogni riga (fila orizzontale) c'è esattamente un abete di ogni altezza;
- in ogni colonna (fila verticale) c'è esattamente un abete di ogni altezza;
- i cartelli con il numero di abeti visibili sono posizionati intorno al campo 3×3.

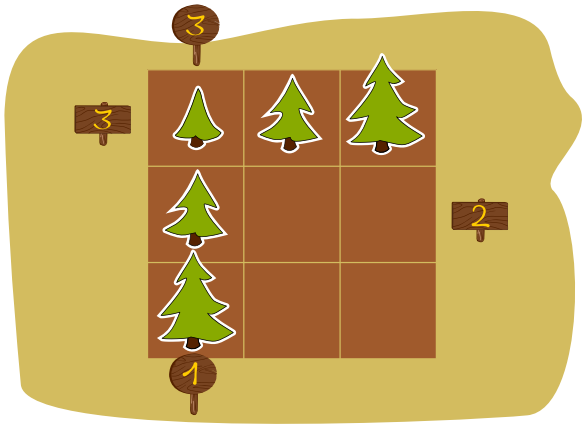
Scrivi in ogni campo l'altezza dell'albero corrispondente.





Soluzione

Sul campo, due cartelli indicano che da quelle posizioni si possono vedere tre abeti. Tutti e tre gli abeti in fila possono essere visti solo se gli abeti sono disposti in modo tale che la loro altezza aumenti, cioè da questa posizione. Questo determina la colonna a sinistra e la riga superiore:

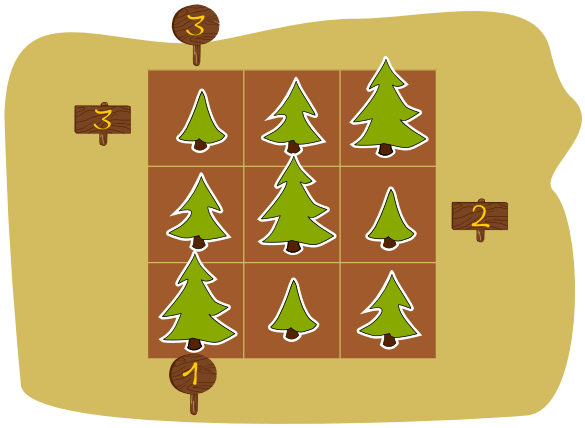


Il cartello a destra con il 2 richiede che da lì siano visibili due abeti, quindi deve esserci un abete di altezza 3 proprio al centro e questa fila centrale è quindi 2 (), 3 (), 1 ().

Gli altri campi sono compilati seguendo la regola del «Sudoku» secondo la quale deve esserci esattamente un abete per ogni altezza in ogni fila.

Al centro della fila inferiore deve esserci un abete di altezza 1 () perché le altre due altezze nella colonna centrale sono già assegnate. Infine, un abete di altezza 2 () deve essere posizionato in basso a destra per completare la fila.

La soluzione completa si presenta così:



Questa è l'informatica!

Questo compito si concentra su tre competenze di base degli informatici.

La prima è quella di trovare una soluzione che rispetti determinati vincoli o di correggere una soluzione proposta, se necessario.

In secondo luogo, si tratta della capacità di ricostruire gli oggetti a partire da informazioni parziali sulla loro rappresentazione. Questo è legato alla generazione di oggetti (rappresentazioni di oggetti) a partire dalle limitate informazioni disponibili, se si conosce la regolarità degli oggetti. Tali procedure possono essere utilizzate anche per la compressione di dati.

In terzo luogo, tali campi ad albero con cartelli possono essere utilizzati per generare codici autoverificanti. Gli errori che si verificano durante l'immissione o il trasporto delle informazioni possono essere rilevati o persino corretti automaticamente.

Parole chiave e siti web

- Sudoku
- Rilevazione e correzione d'errore:
  `https://it.wikipedia.org/wiki/Rilevazione_e_correzione_d'errore`

4. Castoro al castello

Un castoro intelligente ha bisogno di un abete per costruire una diga nel fiume. Purtroppo ha solo una carota. Oggi è giorno di mercato nel castello e il castoro vuole scambiare la sua carota con un abete.

Ogni stanza del castello offre due offerte di scambio. La tabella mostra queste offerte:

Stanza A:		→		oppure		→	
Stanza B:		→		oppure		→	
Stanza C:		→		oppure		→	
Stanza D:		→		oppure		→	
Stanza E:		→		oppure		→	
Stanza F:		→		oppure		→	
Stanza G:		→		oppure		→	
Stanza H:		→		oppure		→	

Per esempio, nella stanza B, il castoro può ottenere un cono per un anello, ma non viceversa.

In quale ordine il castoro intelligente deve attraversare le stanze per possedere finalmente l'abete desiderato?

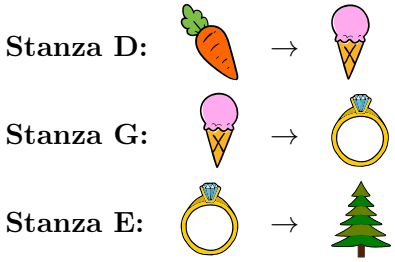
A) DGE: Prima la stanza D, poi la stanza G e infine la stanza E.
B) GGE: Prima la stanza G, poi di nuovo la stanza G e infine la stanza E.
C) AGE: Prima la stanza A, poi la stanza G e infine la stanza E.
D) DBC: Prima la stanza D, poi la stanza B e infine la stanza C.

Soluzione

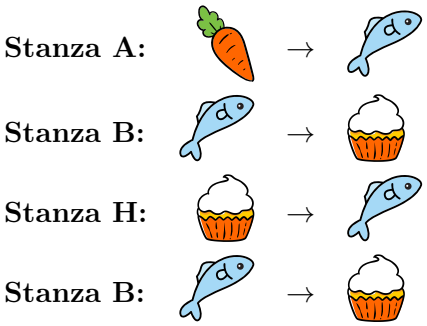
La risposta corretta è A) DGE: prima la stanza D, poi la stanza G e infine la stanza E.

Nella stanza D il castoro scambia la sua carota con un cono. Poi va nella stanza G, dove scambia il cono con un anello. Alla fine il castoro va nella stanza E per scambiare l'anello con un abete.

Questa sequenza complessiva si presenta così:



Per trovare un ordine adeguato delle stanze, due diverse strategie sono utili. La prima strategia cerca di considerare tutte le possibilità di scambio. Si inizia con il primo scambio, dove si può scambiare la carota in cinque stanze (A, D, E, G e H) con 6 oggetti diversi. In seguito, tutte le possibilità di scambio per questi 6 oggetti vengono nuovamente considerate. Questo è complesso e si può anche girare in cerchio, come nell'esempio seguente, dove il castoro può visitare le stanze B e H tutte le volte che vuole:

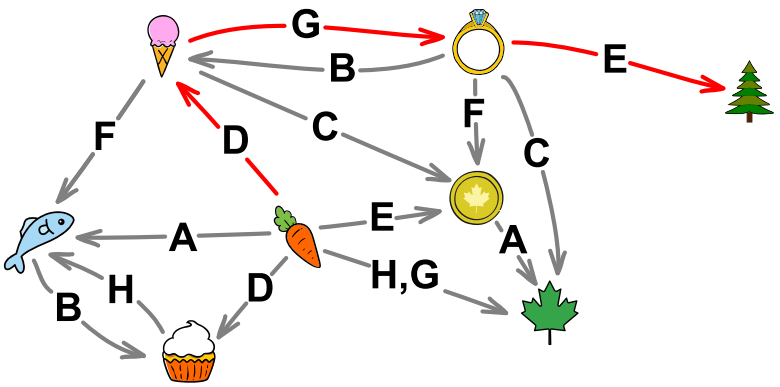


Pertanto questa prima strategia è molto complessa e solo con un po' di fortuna si può avere successo in breve tempo.

La seconda strategia porta rapidamente all'obiettivo in questo compito concreto. Si basa sull'avvio della ricerca dall'obiettivo desiderato, l'abete. Solo nella stanza E il castoro può ottenere l'abete desiderato e l'abete può essere ottenuto solo in cambio di un anello. Il prossimo oggetto desiderato è quindi un anello! L'anello può essere ottenuto solo in una stanza, la stanza G, in cambio di un cono. È possibile ottenere il cono sia nella stanza B scambiandolo con un anello, che nella stanza D scambiandolo con una carota. Quindi il castoro intelligente deve iniziare il suo scambio nella stanza D.

Per una migliore visione d'insieme, la tabella degli scambi possibili può essere visualizzata sotto forma di grafo orientato con archi. Ogni nodo del grafo rappresenta un oggetto di scambio e ogni

arco in uscita rappresenta una possibilità di scambio. Inoltre, l'arco indica lo spazio in cui esiste questa possibilità di scambio.



Questa rappresentazione visiva degli oggetti di scambio, delle possibilità di scambio e delle stanze permette di scoprire facilmente come arrivare dalla carota all'abete, ovvero su un percorso nel grafo orientato: Prima la stanza D, poi la stanza G e infine la stanza E.

Questa è l'informatica!

I *processi di calcolo* possono essere considerati a livello generale come *conseguenze di trasformazioni* (qui si tratta di processi di scambio) o, in modo equivalente, come conseguenze di *stati* di un sistema. Lo stato iniziale del sistema nel nostro caso è la carota e la trasformazione (*la transizione*) da carota a cono cambia questo stato in cono.

Una transizione porta così da uno stato all'altro. Una sequenza di transizioni è anche chiamata *calcolo*.

Questo compito si occupa quindi anche di calcoli a livello molto generale. In questo caso, il sistema *non è deterministico*; ciò significa che ci sono a volte diverse possibili fasi di calcolo, cioè, come nel compito, diversi possibili scambi. Il non determinismo è un altro importante concetto di modellazione nell'informatica. Le possibili fasi di calcolo sono descritte da *regole di trasformazione* (la tabella con possibilità di scambio). Determinare se il castoro può scambiare una carota con un abete, cioè se un certo stato obiettivo del sistema può essere raggiunto da un certo stato di partenza, è il famoso STCON (*st-connettività*) con numerose applicazioni.

Il compito di cui sopra mostra che a volte è una buona idea cercare lo stato di partenza dallo stato di destinazione invece che il contrario. Questa strategia è chiamata anche *ricerca inversa*.

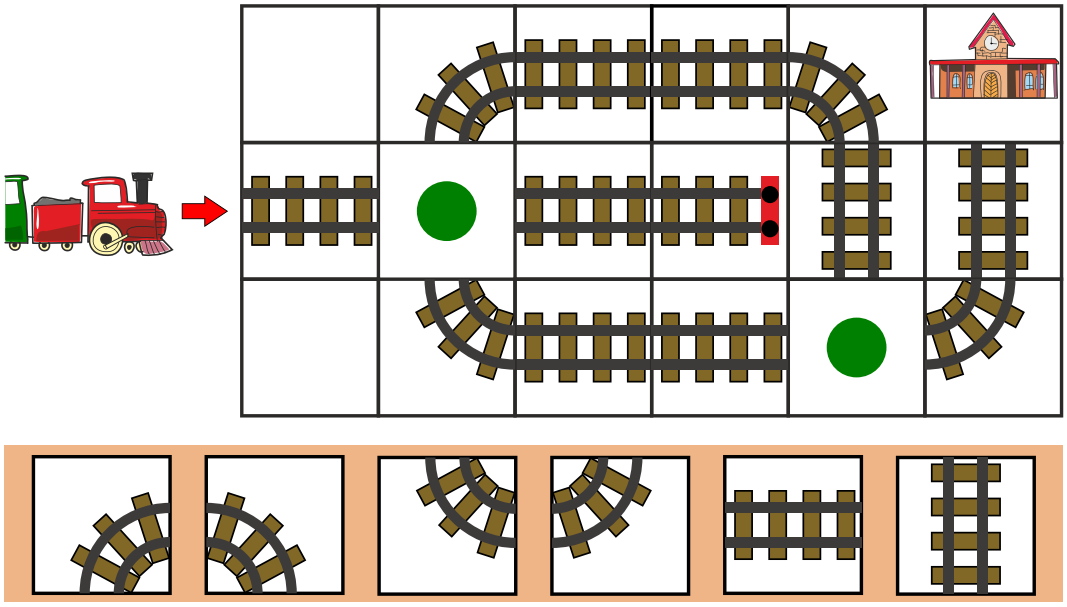
Quando si confrontano le diverse strategie di soluzione, si può notare che il grafo orientato è un modo illustrativo di rappresentare un cosiddetto *spazio di stato* di un sistema con tutte le possibili transizioni tra gli stati. In questo modello di base si potrebbero affrontare i ben noti *algoritmi di ricerca* di base nei grafi, cioè la *ricerca in ampiezza* e la *ricerca in profondità*.

Parole chiave e siti web

- Grafi: `https://it.wikipedia.org/wiki/Grafo`
- Ricerca in profondità: `https://it.wikipedia.org/wiki/Ricerca_in_profondità`
- Ricerca in ampienzza: `https://it.wikipedia.org/wiki/Ricerca_in_ampiezza`

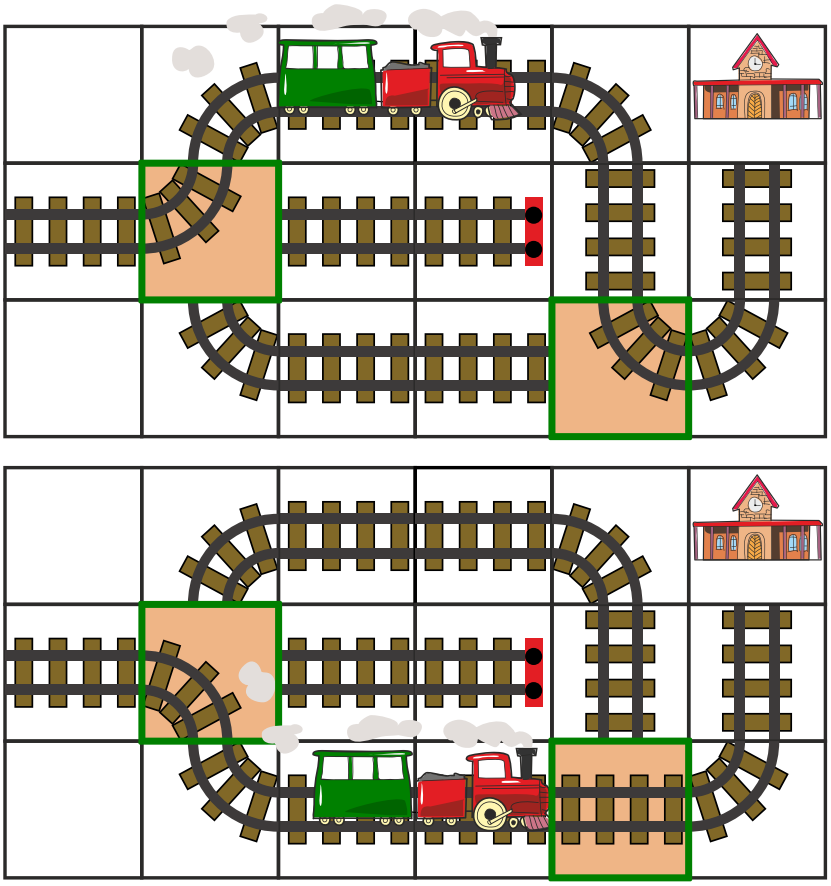
5. Prossima fermata, stazione!

Scegli i binari corretti da mettere nei campi con il punto verde affinché il treno possa raggiungere la stazione.



Soluzione

Per questo problema ci sono le seguenti 2 soluzioni:



Con altre combinazioni il treno deraglia o si scontra con il paraurti.

Questa è l'informatica!

Proprio come un treno viaggia ostinatamente sui binari, un computer esegue ostinatamente le istruzioni di un programma. Non è in grado di riconoscere quando il programma contiene un errore e può bloccarsi, così come un treno può deragliare se i binari sono costruiti in modo errato. Quindi, quando si scrive un programma, bisogna stare molto più attenti che, per esempio, spiegare la strada per la stazione a una persona.

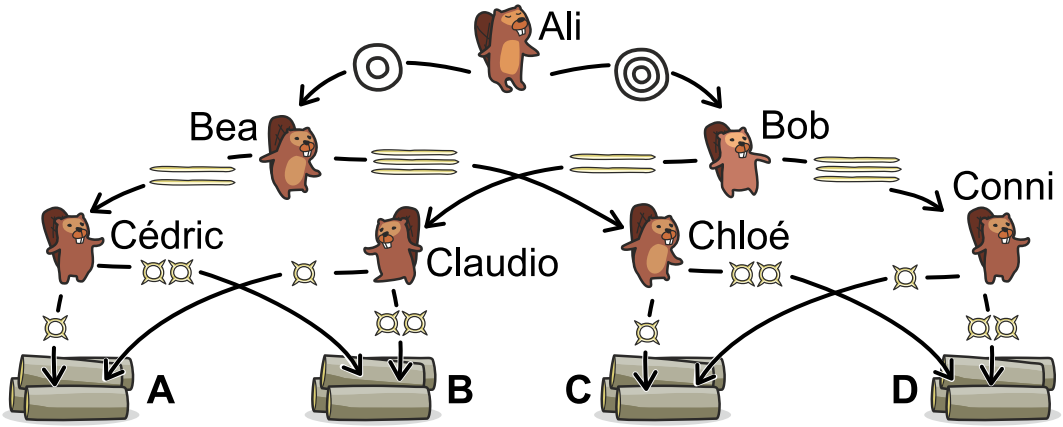
Il compito di cui sopra consiste nell'inserire i comandi mancanti nei punti giusti di un programma in modo da raggiungere l'obiettivo.

Parole chiave e siti web

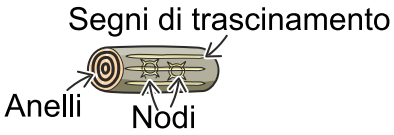
- Programma
- Instruzione: `https://it.wikipedia.org/wiki/Istruzione_(informatica)`
- `https://it.wikipedia.org/wiki/Algoritmo`

6. Tronchi e pile

Nel villaggio dei castori i tronchi sono divisi in quattro gruppi (A, B, C, D) secondo tre caratteristiche (numero di anelli del tronco, segni di trascinamento sulla corteccia e numero di nodi). Il seguente diagramma di decisione mostra come si dividono fra i gruppi.

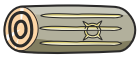


Per esempio, questo tronco viene inserito nella pila D in seguito alle seguenti decisioni:



- Ali vede tre anelli e dà il tronco a Bob;
- Bob vede tre segni di trascinamento e dà il tronco a Conni;
- Conni vede due nodi e mette il tronco sulla pila D.

Su quale pila è collocato questo tronco?



A) Pila A
B) Pila B
C) Pila C
D) Pila D

Soluzione

La risposta corretta è la pila C. Questo perché Ali vede due anelli e dà il tronco a Bea. Bea vede tre segni di trascinamento e dà il tronco a Chloé. Chloé vede un nodo e mette il tronco sulla pila C.

Se lo si desidera, è possibile determinare per ogni pila quali tronchi appartengono alla rispettiva pila. Ci sono due tipi di tronchi su ogni pila.

Sulla pila **A**:

- Tronchi con 2 anelli, 2 segni di trascinamento e 1 nodo.
- Tronchi con 3 anelli, 2 segni di trascinamento e 1 nodo.

Sulla pila **B**:

- Tronchi con 2 anelli, 2 segni di trascinamento e 2 nodi.
- Tronchi con 3 anelli, 2 segni di trascinamento e 2 nodi.

Sulla pila **C**:

- Tronchi con 2 anelli, 3 segni di trascinamento e 1 nodo.
- Tronchi con 3 anelli, 3 segni di trascinamento e 1 nodo.

Sulla pila **D**:

- Tronchi con 2 anelli, 3 segni di trascinamento e 2 nodi.
- Tronchi con 3 anelli, 3 segni di trascinamento e 2 nodi.

Questa è l'informatica!

Questo compito tocca diversi concetti dell'informatica.

In primo luogo, viene affrontato il concetto di diagrammi decisionali, che hanno applicazioni molto versatili nel campo dell'informatica. Qui vengono utilizzati per classificare gli oggetti in categorie selezionate (molto spesso si tratta di alberi di decisione, un tipo speciale di diagrammi decisionali. Il diagramma decisionale del compito non è un albero decisionale in questo caso, perché al livello più basso due gruppi sono posizionati sulla stessa pila).

Qui si può anche pensare al diagramma decisionale come a una rappresentazione astratta dei valori di una funzione di diverse variabili. A livello terminologico, in informatica si parla di «branching programs» (inglese per «programmi di ramificazione»).

Si riferisce anche al concetto di attributi (caratteristiche o proprietà) degli oggetti. Qui gli oggetti hanno tre attributi (anelli del tronco, segni di trascinamento, nodi), con ogni attributo che ha due possibili valori (due o tre anelli annuali o segni di trascinamento e uno o due nodi).

Ci sono molte possibili applicazioni per tali diagrammi decisionali. Uno di questi è la classificazione dei pacchetti di dati quando vengono inviati attraverso una rete (con router o switch).

Parole chiave e siti web

- Albero di decisione: `https://it.wikipedia.org/wiki/Albero_di_decisione`

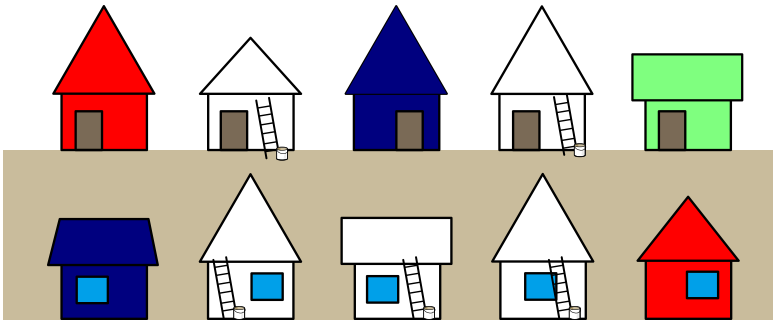
7. Case colorate

Gli abitanti di una strada vogliono dipingere con dei colori le loro case bianche. Ogni casa dovrebbe avere uno dei tre colori: verde chiaro, rosso o blu scuro. Le seguenti regole si applicano per evitare di sembrare noioso:

- Due case che si trovano direttamente l'una accanto all'altra non devono avere lo stesso colore.
- Due case che si trovano direttamente l'una di fronte all'altra non devono avere lo stesso colore.

Alcuni residenti hanno già dipinto le loro case a colori. I restanti residenti devono ora dipingere le loro case in modo che le regole non vengano violate.

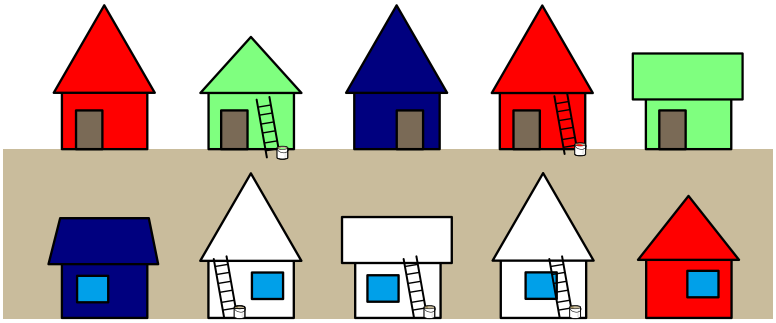
Trova i colori corrispondenti per i residenti.



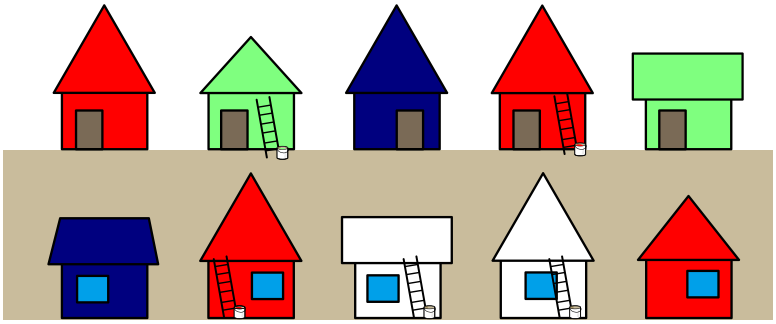
Soluzione

Il modo più semplice per scoprire i colori delle case è scoprirli passo dopo passo.

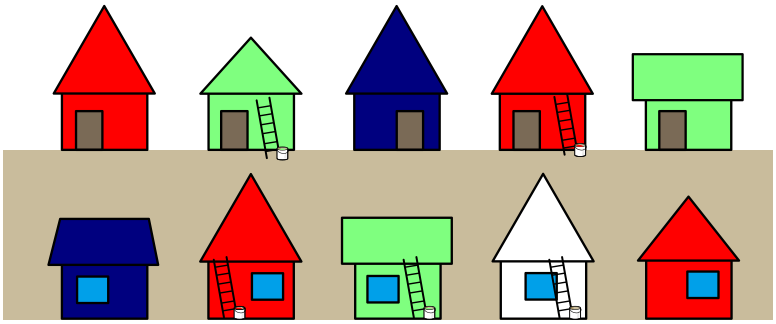
Le due case bianche sul lato superiore della strada sono circondate da due case di colore diverso a sinistra e a destra. Pertanto, possono essere dipinti in un solo colore particolare senza infrangere le regole: la casa bianca in alto a sinistra in verde chiaro e la casa bianca in alto a destra in rosso.



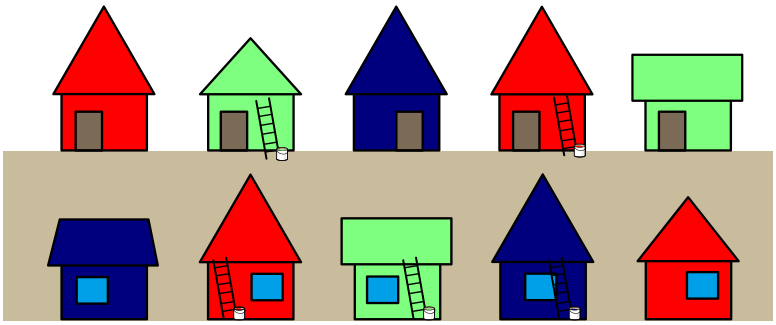
Poi si vede che la casa bianca in basso a sinistra deve essere dipinta di rosso perché la casa direttamente a sinistra è blu scuro e la casa di fronte è verde chiaro:



Quasi la stessa considerazione può essere fatta ora per la casa di mezzo sul lato inferiore della strada: Deve essere dipinta di verde chiaro, perché direttamente alla sua sinistra c'è la casa appena dipinta di rosso e di fronte c'è una casa blu scuro.



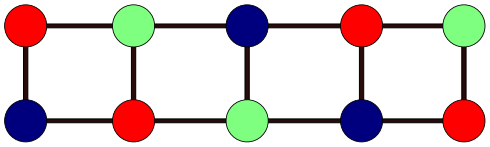
Infine, si può anche scegliere il colore per la casa bianca sul lato destro della strada: La casa direttamente a destra e la casa di fronte sono entrambe rosse, ma poiché la casa direttamente a sinistra è verde chiaro, l'unica opzione rimasta è dipingere la casa di blu scuro:



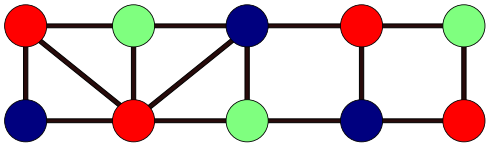
Questa è l'informatica!

In termini astratti, questo compito consiste nel trovare una soluzione che soddisfi le restrizioni date (regole). Questo è un problema molto comune nell'informatica.

Le case e i loro vicini diretti (sia a sinistra che a destra e dall'altra parte della strada) possono essere modellati bene con l'aiuto di un *grafo*, una struttura di dati ampiamente utilizzata nell'informatica. Ogni casa è un *nodo* e ogni vicinato diretto è un *arco*:



Nell' immagine, i nodi sono già colorati come le case corrispondenti. Per le case c'era la regola che le case vicine dovevano avere colori diversi. Nell' immagine la colorazione dei nodi è quindi tale che i nodi collegati direttamente da un arco non hanno mai lo stesso colore. Che ci sia una colorazione valida del grafo con tre colori non è ovvio. Se si aggiungono due archi come nell'immagine successiva, non c'è più una colorazione valida: non importa come si distribuiscono i tre colori in questo grafo, ci sono sempre due nodi direttamente collegati con lo stesso colore.



Ma con quattro colori funziona di nuovo. Forse funziona sempre con quattro colori? La risposta è di nuovo no. Ma almeno un tipo di grafi può sempre essere colorato con quattro colori: i cosiddetti *grafi planari*. Si tratta di grafi che possono essere disegnati in modo che nessun arco si incroci. (Il grafo nell'ultima immagine non è planare, cioè a causa delle connessioni dei quattro nodi all'estrema sinistra). Che i grafi planari hanno una colorazione valida con quattro colori è chiamato il *teorema dei quattro colori*.

Il teorema dei quattro colori è particolarmente interessante per la creazione di mappe. Se si immagina ogni paese come un nodo e poi si collegano i paesi vicini con un arco, si ottiene sempre un grafo planare. (A rigor di termini, per questo dobbiamo escludere l'esistenza delle cosiddette enclavi ed esclavi, cioè parti di un paese che si trovano completamente in un altro paese). Questo grafo può quindi essere colorato con quattro colori validi, e quindi anche i paesi corrispondenti sulla mappa possono essere colorati con quattro colori, in modo che i paesi vicini abbiano sempre colori diversi.

La prova che quattro colori sono sufficienti non è facile. Che cinque colori siano sufficienti era già noto 200 anni fa. I matematici Kenneth Appel e Wolfgang Haken hanno dimostrato nel 1976 che quattro colori sono sufficienti. Hanno usato un computer per controllare un gran numero di eccezioni e controesempi. Ci sono volute più di mille ore al computer per farlo. Controllare tutto a mano sarebbe stato del tutto impossibile. Molti matematici si sono poi chiesti se una tale prova sia valida, perché bisogna fidarsi del computer.

Parole chiave e siti web

- Teorema dei quattro colori: `https://it.wikipedia.org/wiki/Teorema_dei_quattro_colori`
- Grafo: `https://it.wikipedia.org/wiki/Grafo`

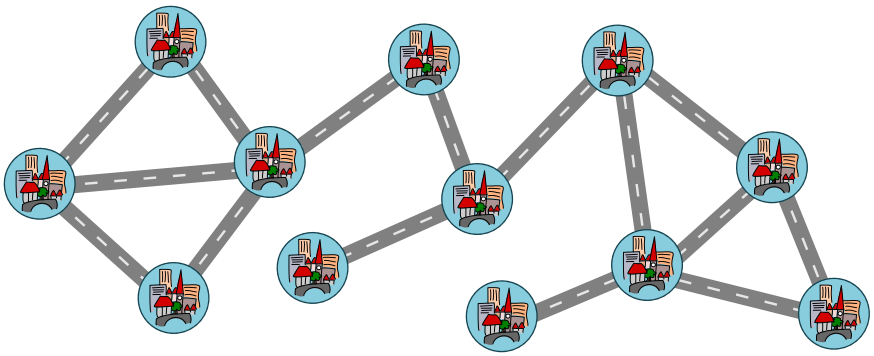
8. Considerazioni epidemiologiche

Biberland è composta da 12 città, che sono collegate da strade. Le città che sono direttamente o indirettamente collegate da strade formano una comunità commerciale. La mappa mostra quindi nella sua forma attuale un'unica comunità commerciale di 12 città.

Per contenere un'epidemia, il traffico deve essere ridotto. Il parlamento di Biberland decide di chiudere esattamente due strade per dividere le città in tre comunità commerciali separate.

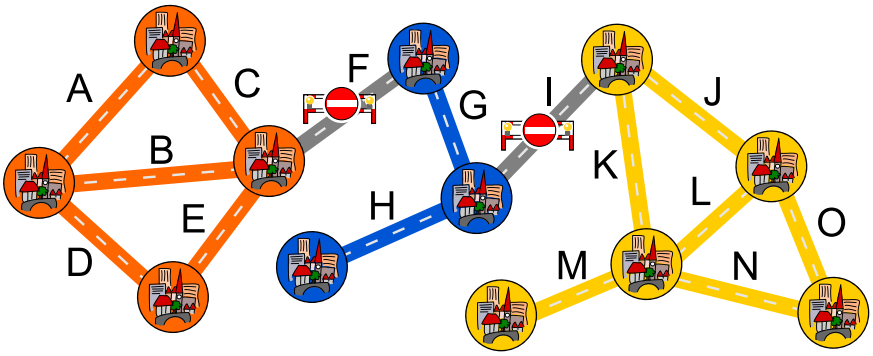
Per non isolare nessuno più del necessario, la più piccola comunità commerciale dovrebbe essere composta dal maggior numero possibile di città.

Quali due strade dovrebbero essere chiuse? Indicale.



Soluzione

La risposta corretta è: le strade F e I nella foto qui sotto devono essere bloccate. Questo crea comunità commerciali di 3, 4 e 5 città.



E' ovvio che dobbiamo guardare solo alle strade che, se sono bloccate, causeranno anche la divisione della comunità commerciale perché sono l'unico collegamento. Dopo tutto, abbiamo bisogno di due vere e proprie divisioni per raggiungere tre unità. Ad esempio, non ha senso chiudere la strada B, perché si possono comunque raggiungere tutte le città via A e C. Rimangono quindi solo le candidate F, G, H, I e M per il blocco.

Se si provano tutte le 10 possibilità di chiudere due delle cinque strade, si otterrà la risposta di cui sopra. Come essere umano, si vede subito che il blocco H o M taglierebbe solo una singola città ed è quindi fuori questione. Ciò limita ulteriormente il numero di possibilità da prendere in considerazione.

Questa è l'informatica!

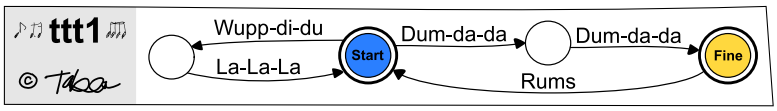
Nell'informatica, una data rete è spesso suddivisa in cosiddette *componenti connesse*. In una componente connessa, tutte le parti sono collegate tra loro attraverso percorsi diretti o indiretti, mentre non c'è alcun collegamento tra le diverse componenti connesse. Ovviamente, l'applicazione è in reti di computer dove è rilevante quali computer possono essere raggiunti da quali altri. Ma anche, ad esempio, nel riconoscimento ottico dei caratteri (OCR), è importante sapere quali punti sono «connessi».

Parole chiave e siti web

- Componente connessa:
  `https://it.wikipedia.org/wiki/Componente_connessa_(teoria_dei_grafi)`

9. Il ritmo di Tabea

Tabea ha molto successo nel creare testi di canzoni con il marchio ttt: Tabea's Tactful Texts. I testi possono essere prodotti con il seguente diagramma ttt1:



Per produrre una canzone, Tabea inizia da «Start» (Start) e segue una delle frecce in uscita. Se ci sono diverse possibilità, può scegliere quella che preferisce. Canta le sillabe corrispondenti lungo il percorso nell'ordine dato. Se raggiunge «Fine» (Fine), la canzone può finire ma può anche continuare.

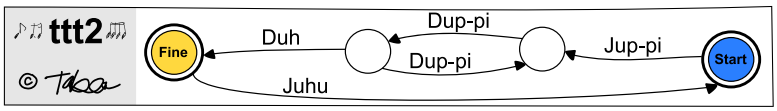
Possibili canzoni possono essere:

«Wupp-di-du La-La-La Wupp-di-du La-La-La
Dum-da-da Dum-da-da Rums Dum-da-da Dum-da-da»

Oppure

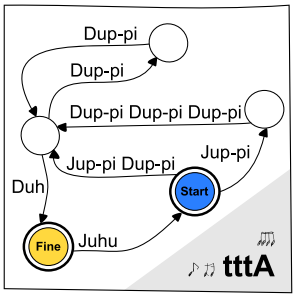
«Dum-da-da Dum-da-da Rums Wupp-di-du La-La-La
Dum-da-da Dum-da-da Rums Wupp-di-du La-La-La
Dum-da-da Dum-da-da Rums Dum-da-da Dum-da-da»

Nel novembre 2020 Tabea produce nuovi testi con il diagramma ttt2:

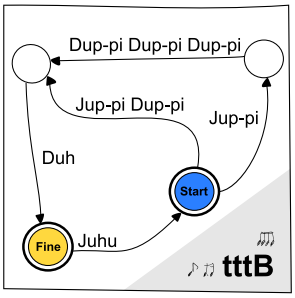


Con quali dei seguenti diagrammi si possono creare esattamente gli stessi testi come con il diagramma ttt2?

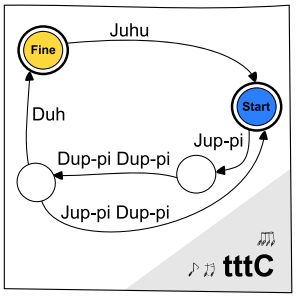
A)



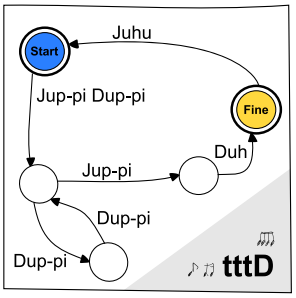
B)



C)

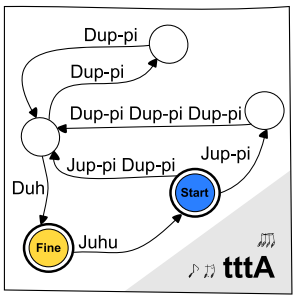


D)

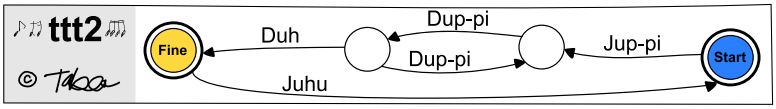


Soluzione

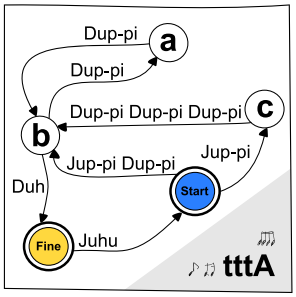
La risposta corretta è A) ossia il diagramma tttA.



Se si produce un brano con il diagramma ttt2, esso inizierà sempre con «Jup-pi» e seguirà almeno un «Dup-pi». Ora continua o direttamente con «Duh» o con un numero pari di ulteriori «Dup-pi» e poi «Duh». Ora la canzone può essere terminata o continuare con un «Juhu» e ricominciare dall'inizio.



Il diagramma tttA raggiunge esattamente lo stesso risultato: Dallo «Start», il brano può partire direttamente in b e quindi con «Jup-pi Dup-pi» o via c con «Jup-pi Dup-pi Dup-pi Dup-pi Dup-pi Dup-pi». Dopo di che, una deviazione tramite a permette di aggiungere un numero pari qualsiasi di «Dup-pi», poi si può usare «Duh» per arrivare alla fine della canzone. Proprio come in ttt2 si può ricominciare dopo «Juhu».



Sia il diagramma ttt2 che il diagramma tttA possono produrre dopo «Jup-pi» un numero dispari a volontà di «Dup-pi». Il diagramma tttB invece può produrre solo 1 o 3 «Dup-pi» di fila e il diagramma tttC solo 1 o 2. Il diagramma tttD crea invece un numero dispari di Dup-pi, ma precede il «Duh» finale con un altro «Jup-pi», che ttt2 non può creare. Quindi tttA è l'unica risposta possibile.

Questa è l'informatica!

Un compito importante dell'informatica è quello di riconoscere le strutture nei dati, ad esempio nel linguaggio, come il testo di una canzone. I diagrammi rappresentano i cosiddetti automi finiti con i quali si possono definire regole molto severe per la generazione e il riconoscimento di determinate lingue. Questo a sua volta è cruciale nello sviluppo dei linguaggi di programmazione. Nel nostro esempio,

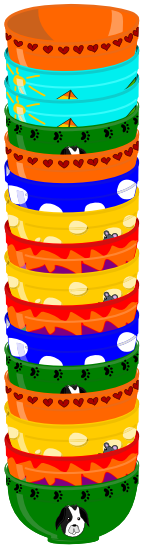
l'automa finito descrive l'insieme di canzoni che possono essere create con esso. Il riconoscimento del modello è importante anche in molti altri settori, come l'elaborazione del linguaggio naturale.

Parole chiave e siti web

- Automi finiti: `https://it.wikipedia.org/wiki/Automa_a_stati_finiti_deterministico`
- Linguaggi formali: `https://it.wikipedia.org/wiki/Linguaggio_formale`
- `https://sites.google.com/isabc.ca/computationalthinking/pattern-recognition`

10. Pila di ciotole

Tre fratelli vogliono mangiare da tre ciotole identiche a colazione. In cucina hanno un'alta pila di ciotole. Per precauzione possono prendere sempre solo una ciotola alla volta dalla cima della pila.

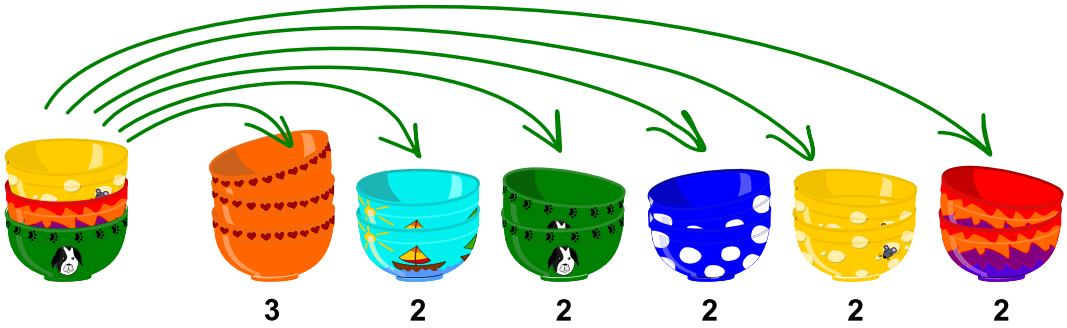


Qual è il numero minore di ciotole che si devono prendere dalla pila mostrata nella foto per averne tre di un tipo?

A) 3 ciotole
B) 4 ciotole
C) 5 ciotole
D) 6 ciotole
E) 7 ciotole
F) 8 ciotole
G) 9 ciotole
H) 10 ciotole
I) 11 ciotole
J) 12 ciotole
K) 13 ciotole
L) 14 ciotole
M) 15 ciotole
N) 16 ciotole

Soluzione

Risposta K): Almeno 13 ciotole devono essere prese dalla pila per ottenere tre ciotole dello stesso tipo.



Questa è l'informatica!

Una *pila*, in informatica spesso chiamata *stack* e a volte struttura *LIFO*, è un modo molto comune di immagazzinare le cose. Uno stack è una struttura molto semplice, ma potente, che viene spesso utilizzata nella programmazione. Ci sono regole su come mettere le cose su una pila e come rimuoverle, di solito solo dall'alto. In questo compito ci occupiamo solo di rimuovere le cose dalla pila. La regola dice che solo l'oggetto più in alto può essere preso dalla pila. Se si vuole ottenere la decima ciotola nella pila, bisogna togliere dieci ciotole una ad una. È importante avere un posto dove poter mettere le altre nove ciotole; questo vale anche per la programmazione. Se abbiamo una seconda pila e possiamo fare le pile alte quanto vogliamo, allora teoricamente potremmo già calcolare con essa tutto ciò che può essere calcolato con un computer! (Nell'informatica, questa proprietà è anche chiamata *Turing equivalenza*.) Queste semplici pile sono davvero potenti!

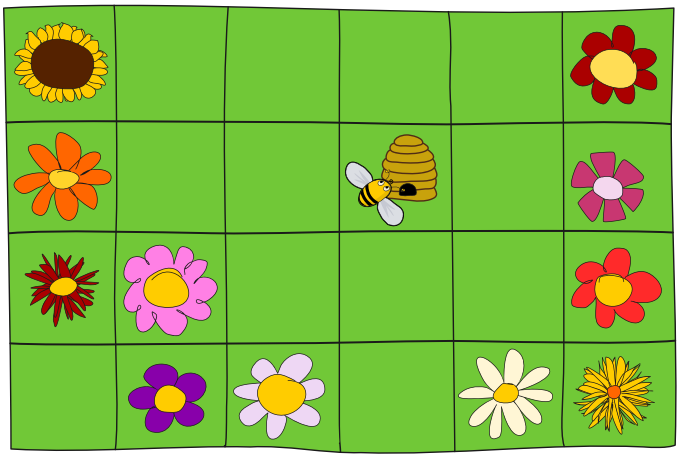
Parole chiave e siti web

- Pila: `https://it.wikipedia.org/wiki/Pila_(informatica)`
- Macchina di Turing: `https://it.wikipedia.org/wiki/Macchina_di_Turing`

11. Dall'alveare ai fiori

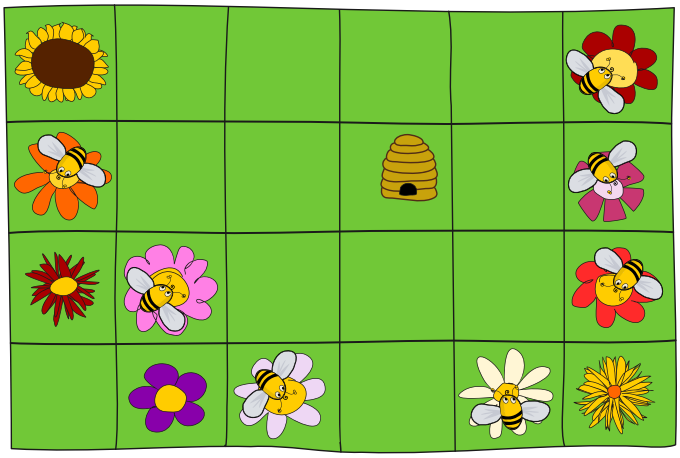
Un'ape vola in su, in giù, a sinistra o a destra. Per volare la distanza di un quadrato ci impiega 10 minuti. Vola dall'alveare, per un massimo di 30 minuti prima di tornare indietro.

Disegna un cerchio attorno ai fiori che si possono raggiungere dall'alveare in massimo 30 minuti.



Soluzione

I fiori con sopra un'ape sono raggiungibili dall'alveare in massimo 30 minuti:



L'immagine sottostante mostra per ogni campo quanti minuti necessita un'ape per raggiungerlo dall'alveare. Così in mezz'ora tutti i campi con un 10, 20 o 30 sono raggiungibili.

Riempire i numeri funziona in questo modo: Nelle caselle accanto all'alveare scriviamo 10 perché l'ape ha bisogno di 10 minuti per volare dall'alveare a lì. Poi scriviamo 20 in tutti i campi vuoti accanto a un campo con 10, perché l'ape ha bisogno di 10 minuti per volare da un campo all'altro. Continueremo a farlo. Così ne scriviamo 30 in tutti i campi vuoti accanto a un campo con 20. Poi ne scriviamo 40 in tutti gli spazi vuoti accanto a uno spazio con 30. Infine, scriviamo 50 in tutti i campi vuoti accanto a un campo con 40.

Questa è l'informatica!

Quando si risolve il compito, si calcola per ogni campo il tempo in cui un'ape può raggiungerlo dall'alveare. Per prima cosa vengono determinati i campi raggiungibili in 10 minuti. Questi vengono poi utilizzati per determinare i campi che si trovano a 20 minuti di distanza. Utilizzando i campi distanti 20 minuti, si trovano poi i campi distanti 30 minuti e così via.

Quindi utilizziamo i risultati già calcolati e memorizzati (i numeri dei campi riempiti) per calcolare ulteriori risultati (i numeri dei campi vicini, ancora vuoti). Questo principio molto generale si chiama *programmazione dinamica*. Di solito è importante l'ordine in cui vengono calcolati i risultati. Anche questo è da considerare con il volo delle api.

Nel compito un'ape vola in 10 minuti solo su, giù, a sinistra o a destra. Questo è un po' insolito, perché in realtà un'ape probabilmente volerebbe anche in diagonale sui campi. Con questa ipotesi più realistica, i campi raggiungibili in mezz'ora sarebbero limitati da un cerchio invece che da un diamante come nel compito.

La consueta misura della distanza che porta ad un cerchio è chiamata distanza euclidea. La misura della distanza così come nel compito in cui si è autorizzati a muoversi solo orizzontalmente o verticalmente attraverso i quadrati è chiamata *distanza di Manhattan* (il nome deriva dalle reti stradali a griglia delle città moderne come Manhattan).

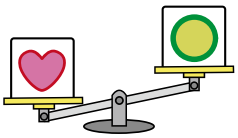
Parole chiave e siti web

- Programmazione dinamica: `https://it.wikipedia.org/wiki/Programmazione_dinamica`
- Distanza euclidea: `https://it.wikipedia.org/wiki/Distanza_euclidea`
- Distanza di Manhattan (o geometria del taxi): `https://it.wikipedia.org/wiki/Geometria_del_taxi`

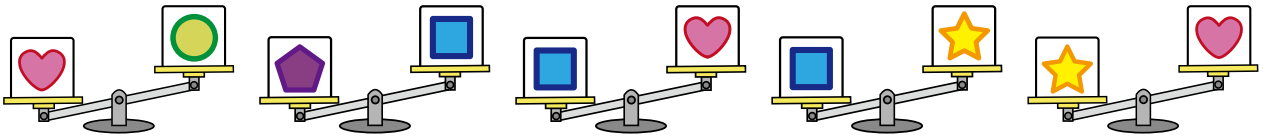
12. Comparazioni pesanti

Cinque scatole sono contrassegnate da cinque diversi simboli:  ,  ,  ,  e  .

Con l'aiuto di una bilancia si comparano due scatole alla volta. La seguente comparazione mostra, ad esempio, che  è più pesante di  :



In totale sono state effettuate cinque comparazioni:



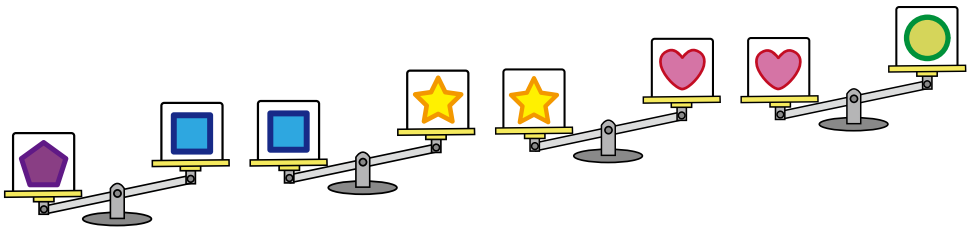
Qual è la scatola più pesante?

A) 
B) 
C) 
D) 
E) 

Soluzione

La scatola C) con il pentagono è la più pesante.

La figura seguente contiene quattro delle cinque comparazioni effettuate e tutte e cinque le scatole.



Così si vede subito: la scatola con il pentagono è più pesante di quella con il quadrato. La scatola con il quadrato è più pesante della scatola con la stella. La scatola con la stella è più pesante di quella con il cuore. E la scatola con il cuore è più pesante della scatola con il cerchio.

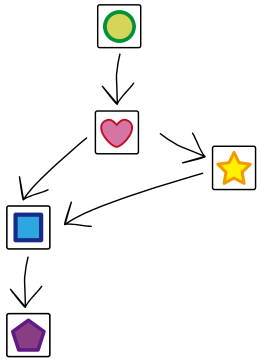
Da questo si può ora concludere che la scatola con il pentagono è più pesante di tutte le altre. Ciò è dovuto ad una speciale proprietà di comparazione dei pesi:

Se A è più pesante di B e B è più pesante di C, allora anche A è più pesante di C. Questa proprietà molto importante si chiama *relazione transitiva*.

A proposito, c'è un modo intelligente per abbreviare questo compito. Poiché si sta cercando la scatola più pesante, è sufficiente cercare la scatola che non è più leggera di qualsiasi altra scatola, e questa è solo la scatola con il pentagono.

Questa è l'informatica!

Fondamentalmente, questo compito riguarda *l'ordinamento* di qualsiasi oggetto. In informatica, per l'ordinamento vengono spesso utilizzati dei *grafi* speciali, che consistono in *nodi* (gli oggetti da ordinare) e *archi* (comparazione di due oggetti). Gli oggetti in questo compito sono le scatole e le comparazioni sono le pesate. Se si disegnano gli archi come frecce che puntano all'oggetto più pesante, il grafo per questo compito si presenta così:



Gli oggetti devono ora essere ordinati in fila, in modo che le frecce partano sempre dagli oggetti più a sinistra verso gli oggetti più a destra. Una tale disposizione è chiamata *ordinamento topologico*. Un ordinamento topologico si ottiene molto semplicemente rimuovendo ripetutamente un nodo dal grafo verso il quale non c'è alcuna freccia puntata e ponendo i nodi rimossi in quest'ordine.

Ma attenzione: non tutti i grafi hanno un ordinamento topologico. Per esempio, non c'è un ordinamento topologico se ci sono tre frecce che puntano in un cerchio.

Parole chiave e siti web

- Relazione transitiva: `https://it.wikipedia.org/wiki/Relazione_transitiva`
- Grafo: `https://it.wikipedia.org/wiki/Grafo`
- Ordinamento topologico: `https://it.wikipedia.org/wiki/Ordinamento_topologico`

A. Autori dei quesiti

Serge Adam
Faisal Al-Sudani
Carlo Bellettini
Linda Björk Bergsveinsdóttir
Daniela Bezáková
Sarah Chan
Marios O. Choudary
Kris Coolsaet
Valentina Dagienė
Christian Datzko
Susanne Datzko
Hanspeter Erni
Lidia Feklistova
Fabian Frei
Gerald Futschek
Jens Gallenbacher
Christian Giang
Juraj Hromkovič
Alisher Ikramov
Tiberiu Iorgulescu
Takeharu Ishizuka
Ungyeol Jung
Vaidotas Kinčius
Sophie Koh
Dennis Komm
Chia-Yi Ku
Regula Lacher
Taina Lehtimäki
Marielle Léonard
Judith Lin
Lynn Liu
Vu Van Luan
Hiroki Manabe
Hamed Mohebbi
Kwangsik Moon
Anna Morpurgo
Xavier Muñoz
Hiroyuki Nagataki
Tom Naughton
Ágnes Erdősné Németh
Gabriel Parriaux
Jean-Philippe Pellet
Margot Phillipps
Wolfgang Pohl
Pedro Ribeiro
Chris Roffey
Peter Rossmanith
Vipul Shah
Maiko Shimabuku
Peter Tomcsányi
Monika Tomcsányiová
Meng-ting Tsai

 Michael Weigend

 Jonas Winckler

B. Sponsoring: concorso 2020

HASLERSTIFTUNG

http://www.haslerstiftung.ch/

http://www.baerli-biber.ch/

verkehrshaus.ch

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

Kanton Zürich
Volkswirtschaftsdirektion
Amt für Wirtschaft und Arbeit

Standortförderung beim Amt für Wirtschaft und Arbeit Kanton Zürich

i-factory (Musée des transports, Lucerne)

UBS

http://www.ubs.com/

OXOCARD

http://www.oxocard.ch/
OXOcard
OXON

educaTEC

https://educatec.ch/
educaTEC

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.

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

`http://www.phlu.ch/`
Pädagogische Hochschule Luzern

`https://www.fhnw.ch/de/die-fhnw/hochschulen/ph`
Pädagogische Hochschule FHNW

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

`https://www.zhdk.ch/`
Zürcher Hochschule der Künste

C. Ulteriori offerte

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.