INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Aufgaben und Lösungen 2020

Schuljahre 5/6

https://www.informatik-biber.ch/

Herausgeber:

Susanne Datzko, Fabian Frei, Juraj Hromkovič,
Regula Lacher, Jean-Philippe Pellet

0101001101010110010010010000001001011010101001101010010011010010010100101010100110101001001001000100101

SV!A

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

Mitarbeit Informatik-Biber 2020

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

Projektleitung: Nora A. Escherle

Herzlichen Dank für die Aufgabenentwicklung für den Schweizer-Wettbewerb an:
Juraj Hromkovič, Michael Barot, Christian Datzko, Jens Gallenbacher, Dennis Komm, Regula Lacher, Peter Rossmanith: ETH Zürich, Ausbildunges- und Beratungszentrum für Informatikunterricht

Die Aufgabenauswahl wurde erstellt in Zusammenarbeit mit den Organisatoren von Bebras in Deutschland, Österreich, Ungarn, Slowakei und Litauen. Besonders danken wir:
Valentina Dagienė: Bebras.org
Wolfgang Pohl, Hannes Endreß, Ulrich Kiesmüller, Kirsten Schlüter, Michael Weigend: Bundesweite Informatikwettbewerbe (BWINF), Deutschland
Wilfried Baumann, Anoki Eischer: Österreichische Computer Gesellschaft
Gerald Futschek, Florentina Voboril: Technische Universität Wien
Zsuzsa Pluhár: ELTE Informatikai Kar, Ungarn
Michal Winzcer: Comenius University, Slowakei

Die Online-Version des Wettbewerbs wurde auf cuttle.org realisiert. Für die gute Zusammenarbeit danken wir:
Eljakim Schrijvers, Justina Dauksaite, Arne Heijenga, Dave Oostendorp, Andrea Schrijvers, Alieke Stijf, Kyra Willekes: cuttle.org, Niederlande
Chris Roffey: University of Oxford, Vereinigtes Königreich

Für den Support während den Wettbewerbswochen danken wir:
Hanspeter Erni: Schulleitung Sekundarschule Rickenbach
Gabriel Thullen: Collège des Colombières
Beat Trachsler: Kantonsschule Kreuzlingen

Christoph Frei: Chragokyberneticks (Logo Informatik-Biber Schweiz)
Dr. Andrea Leu, Maggie Winter, Brigitte Manz-Brunner: Senarclens Leu + Partner AG

Die deutschsprachige Fassung der Aufgaben wurde ähnlich auch in Deutschland und Österreich verwendet.

Die französischsprachige Übersetzung wurde von Elsa Pellet und die italienischsprachige Übersetzung von Christian Giang erstellt.

INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Der Informatik-Biber 2020 wurde vom Schweizerischen Verein für Informatik in der Ausbildung SVIA durchgeführt und von der Hasler Stiftung unterstützt.

HASLERSTIFTUNG

Dieses Aufgabenheft wurde am 9. September 2021 mit dem Textsatzsystem LaTeX erstellt. Wir bedanken uns bei Christian Datzko für die Entwicklung und langjährige Pflege des Systems zum Generieren der 36 Versionen dieser Broschüre (nach Sprachen und Schulstufen). Das System wurde analog zum Vorgänger-System neu programmiert, welches ab 2014 gemeinsam mit Ivo Blöchliger entwickelt wurde. Jean-Philippe Pellet danken wir für die Entwicklung der `bebras` Toolchain, die seit 2020 für die automatisierte Konvertierung der Markdown- und YAML-Quelldokumente verwendet wird.

Hinweis: Alle Links wurden am 1. Dezember 2020 geprüft.

Vorwort

Der Wettbewerb «Informatik-Biber», der in verschiedenen Ländern der Welt schon seit mehreren Jahren bestens etabliert ist, will das Interesse von Kindern und Jugendlichen an der Informatik wecken. Der Wettbewerb wird in der Schweiz in Deutsch, Französisch und Italienisch vom Schweizerischen Verein für Informatik in der Ausbildung SVIA durchgeführt und von der Hasler Stiftung im Rahmen des Förderprogramms FIT in IT unterstützt.

Der Informatik-Biber ist der Schweizer Partner der Wettbewerbs-Initiative «Bebras International Contest on Informatics and Computer Fluency» (`https://www.bebras.org/`), die in Litauen ins Leben gerufen wurde.

Der Wettbewerb wurde 2010 zum ersten Mal in der Schweiz durchgeführt. 2012 wurde zum ersten Mal der «Kleine Biber» (Stufen 3 und 4) angeboten.

Der Informatik-Biber regt Schülerinnen und Schüler an, sich aktiv mit Themen der Informatik auseinander zu setzen. Er will Berührungsängste mit dem Schulfach Informatik abbauen und das Interesse an Fragenstellungen dieses Fachs wecken. Der Wettbewerb setzt keine Anwenderkenntnisse im Umgang mit dem Computer voraus – ausser dem «Surfen» im Internet, denn der Wettbewerb findet online am Computer statt. Für die Fragen ist strukturiertes und logisches Denken, aber auch Phantasie notwendig. Die Aufgaben sind bewusst für eine weiterführende Beschäftigung mit Informatik über den Wettbewerb hinaus angelegt.

Der Informatik-Biber 2020 wurde in fünf Altersgruppen durchgeführt:

- Stufen 3 und 4 («Kleiner Biber»)
- Stufen 5 und 6
- Stufen 7 und 8
- Stufen 9 und 10
- Stufen 11 bis 13

In den Altersklassen 3 und 4 hatten 9 Aufgaben zu lösen, nämlich aus den drei Schwierigkeitsstufen leicht, mittel und schwer jeweils drei. Für die Altersklassen 5 und 6 waren es je vier Aufgaben aus jeder Schwierigkeitsstufe, also 12 insgesamt. Für die restlichen Altersklassen waren es 15 Aufgaben, nämlich fünf Aufgaben pro Schwierigkeitsstufe.

Für jede richtige Antwort wurden Punkte gutgeschrieben, für jede falsche Antwort wurden Punkte abgezogen. Wurde die Frage nicht beantwortet, blieb das Punktekonto unverändert. Je nach Schwierigkeitsgrad wurden unterschiedlich viele Punkte gutgeschrieben beziehungsweise abgezogen:

	leicht	**mittel**	**schwer**
richtige Antwort	6 Punkte	9 Punkte	12 Punkte
falsche Antwort	−2 Punkte	−3 Punkte	−4 Punkte

Dieses international angewandte System zur Punkteverteilung soll den Anreiz zum blossen Erraten der Lösung eliminieren.

Jede Teilnehmerin und jeder Teilnehmer hatte zu Beginn 45 Punkte («Kleiner Biber»: 27 Punkte, Stufen 5 und 6: 36 Punkte) auf dem Punktekonto.

Damit waren maximal 180 Punkte («Kleiner Biber»: 108 Punkte, Stufen 5 und 6: 144 Punkte) zu erreichen, das minimale Ergebnis betrug 0 Punkte.

Bei vielen Aufgaben wurden die Antwortalternativen am Bildschirm in zufälliger Reihenfolge angezeigt. Manche Aufgaben wurden in mehreren Altersgruppen gestellt.

Für weitere Informationen:

SVIA-SSIE-SSII Schweizerischer Verein für Informatik in der Ausbildung
Informatik-Biber
Nora A. Escherle

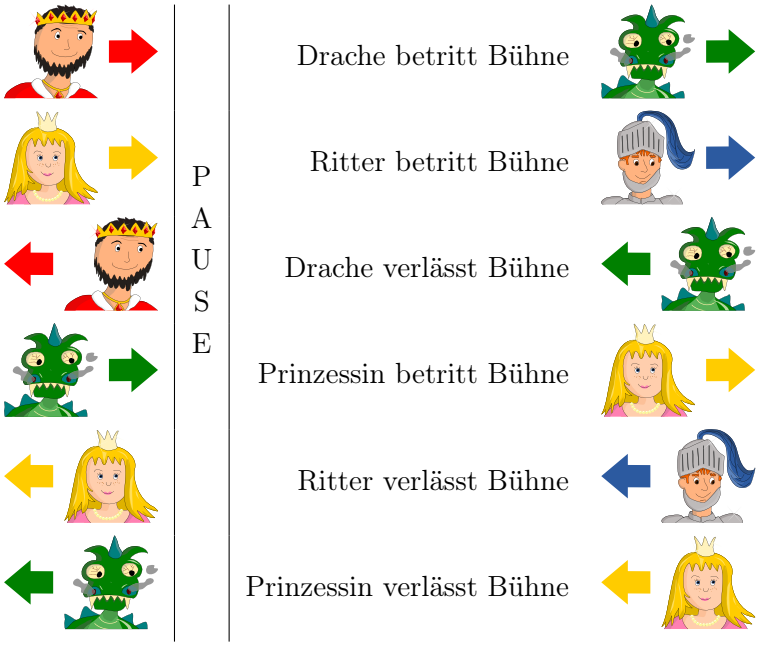
`https://www.informatik-biber.ch/de/kontaktieren/`
`https://www.informatik-biber.ch/`

Inhaltsverzeichnis

1. Das Theaterstück

In einem Theaterstück spielen eine schöne Prinzessin, ein edler Ritter, der weise König und ein böser Drache mit. Am Anfang ist die Bühne leer. Während der Aufführung des Theaterstücks betreten und verlassen diese vier Figuren die Bühne in der folgenden Reihenfolge:

Erster Akt		**Zweiter Akt**
König betritt Bühne		Drache betritt Bühne
Prinzessin betritt Bühne		Ritter betritt Bühne
König verlässt Bühne	P A U S E	Drache verlässt Bühne
Drache betritt Bühne		Prinzessin betritt Bühne
Prinzessin verlässt Bühne		Ritter verlässt Bühne
Drache verlässt Bühne		Prinzessin verlässt Bühne
Pause		**Ende**



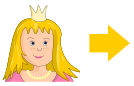
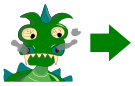
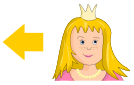
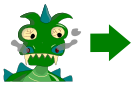
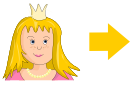
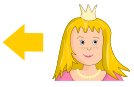
Was wird nicht passieren?

A) Die Prinzessin und der Ritter sind gemeinsam auf der Bühne.
B) Der König und der Drache sind gemeinsam auf der Bühne.
C) Der Ritter betritt die Bühne erst nach der Pause.
D) Der Ritter und der Drache sind gemeinsam auf der Bühne.

Lösung

Die richtige Antwort ist B) «Der König und der Drache sind gemeinsam auf der Bühne.», denn diese Behauptung stimmt während des ganzen Stückes nie.

Man kann sich das schrittweise überlegen:

Handlung	König auf Bühne?	Prinzessin auf Bühne?	Drache auf Bühne?	Ritter auf Bühne?	Übereinstimmung mit Behauptung der Antworten
Erster Akt					
	Ja	Nein	Nein	Nein	
	Ja	Ja	Nein	Nein	
	Nein	Ja	Nein	Nein	
	Nein	Ja	Ja	Nein	
	Nein	Nein	Ja	Nein	
	Nein	Nein	Nein	Nein	
Pause					
Zweiter Akt					
	Nein	Nein	Ja	Nein	
	Nein	Nein	Ja	Ja	C), D)
	Nein	Nein	Nein	Ja	
	Nein	Ja	Nein	Ja	A)
	Nein	Ja	Nein	Nein	
	Nein	Nein	Nein	Nein	
Ende					

Für jede Antwort kann geprüft werden, ob die darin gemachte Behauptung stimmt oder nicht, indem man die Zeilen der Tabellen durchgeht.

Bei der Antwort A) wird nach einer Zeile gesucht, in der sowohl die Prinzessin als auch der Ritter gemeinsam auf der Bühne sind. Das ist in der vierten Zeile des zweiten Aktes der Fall, denn dann betritt die Prinzessin die Bühne, wo der Ritter schon seit der zweiten Zeile ist und bis zur fünften Zeile bleibt. Die Behauptung von Antwort A) stimmt also zu mindestens einem Zeitpunkt.

Bei der Antwort D) wird nach einer Zeile gesucht, in der der Ritter und der Drache gemeinsam auf der Bühne sind. Das ist in der zweiten Zeile des zweiten Aktes der Fall, denn der Ritter betritt die Bühne in der zweiten Zeile, während der Drache die Bühne schon in der ersten Zeile betreten hat und bis zur dritten Zeile bleibt. Die Behauptung von Antwort D) stimmt also zu mindestens einem Zeitpunkt.

Bei der Antwort C) ist die Behauptung von einer anderen Art. Wenn diese stimmen soll, darf der Ritter die Bühne während des gesamten ersten Aktes nicht betreten haben. Hier muss man sich die Spalte des Ritters für den ersten Akt anschauen. Hier steht überall «Nein», also hat der Ritter die Bühne tatsächlich während des gesamten ersten Aktes nicht betreten. Er betritt sie dann aber in der zweiten Zeile des zweiten Aktes, also stimmt die Behauptung von Antwort C) ebenfalls.

Wenn die Behauptung von Antwort B) stimmen würde, müssten der König und der Drache in irgendeiner Zeile gemeinsam auf der Bühne stehen. Doch in keiner der zwölf Zeilen steht in beiden Spalten ein «Ja». Es ist sogar so, dass der König bereits in der dritten Zeile des ersten Aktes die Bühne verlässt und sie bis zum Ende nicht mehr betritt. Der Drache hingegen betritt die Bühne erst in der vierten Zeile des ersten Aktes. Vielleicht begegnen sich die beiden hinter der Bühne, aber auf der Bühne sind sie nie gemeinsam. Damit stimmt die Behauptung von Antwort B) nicht. Also ist B) die korrekte Antwort.

Dies ist Informatik!

Auch wenn man sich aufgrund des Ablaufs des Theaterstücks lebhaft eine Geschichte vorstellen kann, ist in dieser Aufgabe für jede Figur immer nur eine Eigenschaft wichtig: Befindet sie sich zu einem bestimmten Zeitpunkt auf der Bühne oder nicht? Dieses Einschränken des Blicks auf bestimmte Eigenschaften nennt man *Abstraktion*.

In der Informatik kann man solche Abstraktionen sehr gut formulieren. Für jede der vier Figuren definieren wir eine sogenannte *Variable*, die uns die Frage beantwortet, ob sich die Figur gerade auf der Bühne befindet. Die vier Variablen sind: «König auf Bühne?», «Prinzessin auf Bühne?», «Drache auf Bühne?» und «Ritter auf Bühne?». Während des Stückes ändern sich die Antworten auf diese Fragen immer wieder; für jede Frage ist die Antwort manchmal «ja» und manchmal «nein». In der Informatik nennen wir die aktuelle Antwort auf eine Frage den aktuellen *Wert* der zugehörigen Variablen. Der Wert einer Variablen kann sich in der Informatik also immer wieder ändern. (In der Mathematik ist das anders, dort ändern Variablen ihre Werte nicht über die Zeit hinweg.) Die Tabelle in der Antworterklärung zeigt die vier Variablen und die zugehörigen Werte zu jedem Zeitpunkt.

Es gibt noch eine andere Weise, das Theaterstück zu betrachten. Zu jedem Zeitpunkt schauen wir, welche Figuren gerade auf der Bühne sind. (Wir betrachten also die momentanen Werte der vier Variablen.) Jede mögliche Kombination von Figuren nennen wir einen *Zustand* der Bühne. Wenn

eine Figur die Bühne betritt oder verlässt, ändert sich also der Zustand der Bühne. Wir nennen das dann auch einen *Übergang* der Bühne von einem Zustand in einen anderen. Wenn man ein Blatt Papier nimmt und sich für jeden möglichen Zustand (also jede Figurenkombination) einen separaten Kreis zeichnet, kann man das Ganze als eine Abstraktion der Bühne betrachten.

Zusätzlich kann man noch die möglichen Übergänge als Pfeile einzeichnen, die von einem Zustand zu einem anderen führen. Wenn wir das auch noch tun, haben wir das, was wir in der Informatik den *Zustandsgraphen* der Bühne nennen würden.

Zu Beginn des Theaterstückes ist die Bühne leer. Den entsprechenden Zustand nennen wir deshalb *Anfangszustand*. Den Ablauf des Theaterstückes können wir jetzt als einen Weg im Zustandsgraphen einzeichnen. Der Weg beginnt im Anfangszustand und geht dann denjenigen Pfeilen entlang, die der Handlung entsprechen.

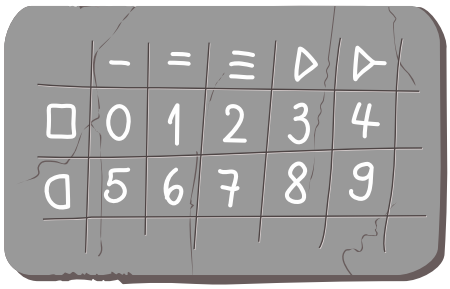
Zustandsgraphen sind in der Informatik sehr wichtig. Bei fast jedem komplizierten System muss man irgendwann über den Zustandsgraphen nachdenken. Für Menschen ist es aber oft sehr mühsam, mit solchen abstrakten Zuständen und Übergängen zu arbeiten. Computer können das hingegen extrem gut. Daher lohnt es sich, wenn Menschen ihre Probleme so mit Zustandsgraphen darstellen können, dass Computer sie dann lösen können.

Stichwörter und Webseiten

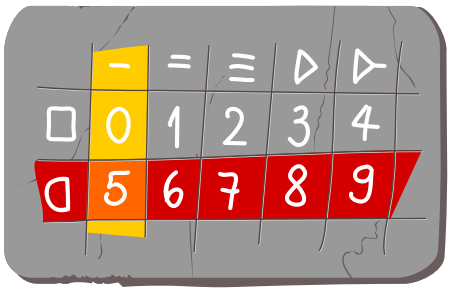
- Variablen: `https://de.wikipedia.org/wiki/Variable_(Programmierung)`
- Zustände, Übergänge, Zustandsgraph:
  `https://de.wikipedia.org/wiki/Zustandsübergangsdiagramm`

2. Baujahr der Biberburg

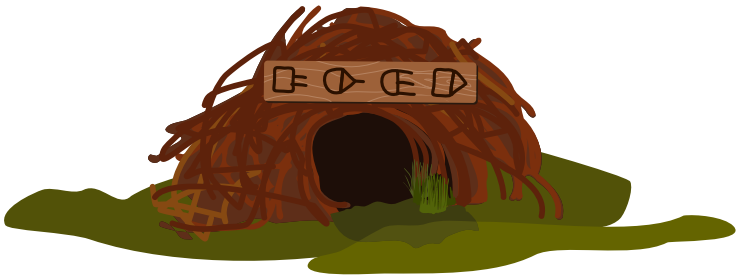
Auf dem Schild über dem Eingang jeder Biberburg steht das Baujahr. Die Biber verwenden für die Ziffern eigene Zeichen. Die Tabelle rechts zeigt, wie man aus den Ziffern die Zeichen der Biber zusammensetzen kann:



Beispielsweise setzen die Biber die Ziffer «5» so zu dem neuen Zeichen  zusammen:



So sieht Cleverias Biberburg aus:



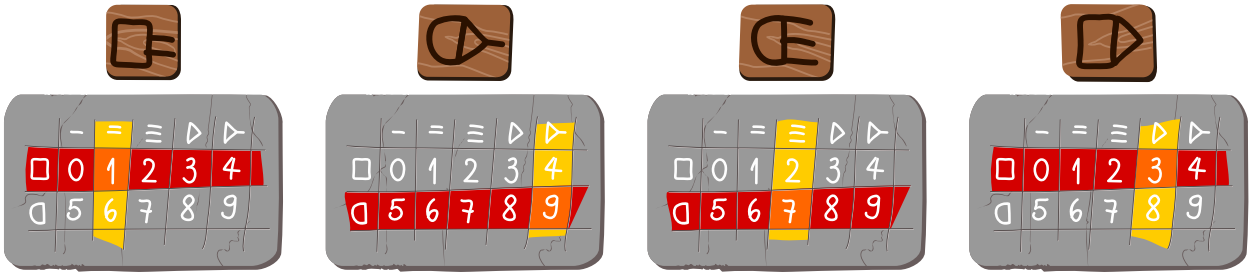
In welchem Jahr wurde Cleverias Biberburg gebaut?

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

Lösung

Das Baujahr der Biberburg findest du heraus, indem du für jedes Symbol die entsprechende Zeile und die entsprechende Spalte bestimmst. An der Kreuzung der Spalte und der Zeile findest du die gesuchte Ziffer.

Weil es vier Symbole sind, machst du das vier Mal.



Die vier Ziffern in der richtigen Reihenfolge ergeben die Zahl 1973.

Dies ist Informatik!

Das Geheimhalten von Informationen und das Schützen von Daten ist eine 4000 Jahre alte Aufgabe. Unzählige Geheimsprachen wurden zu diesem Zweck entwickelt und benutzt. Heute ist Datensicherheit eines der Kernthemen der Informatik. Eine der Methoden, Daten vor unbefugtem Lesen zu schützen, ist sie *zu chiffrieren*. Das Chiffrieren verwandelt einen *Klartext* in einen *Geheimtext*. Das Rekonstruieren des Klartextes aus dem Geheimtext nennt man *Dechiffrieren*. Die Lehre der Geheimschriften nennt man *Kryptologie*.

Die antiken Kulturen verwendeten meistens Geheimschriften, die durch Codierung von Buchstaben mit anderen Buchstaben oder ganz neuen Zeichen erzeugt worden sind. Die Geheimschrift hier ist speziell für den Informatik-Biber entwickelt worden, basiert aber auf einem Konzept aus dem antiken Palästina. Die damalige Sicherheitsregel war, dass nur Geheimschriften verwendet worden sind, die man leicht auswendig lernen kann. Eine schriftliche Beschreibung der Geheimschrift aufzubewahren, betrachtete man als zu grosses Risiko. Eine Tabelle, wie sie hier verwendet wird, kann man gut auswendig lernen. Die berühmte Geheimschrift der Freimaurer basiert auf diesem Prinzip.

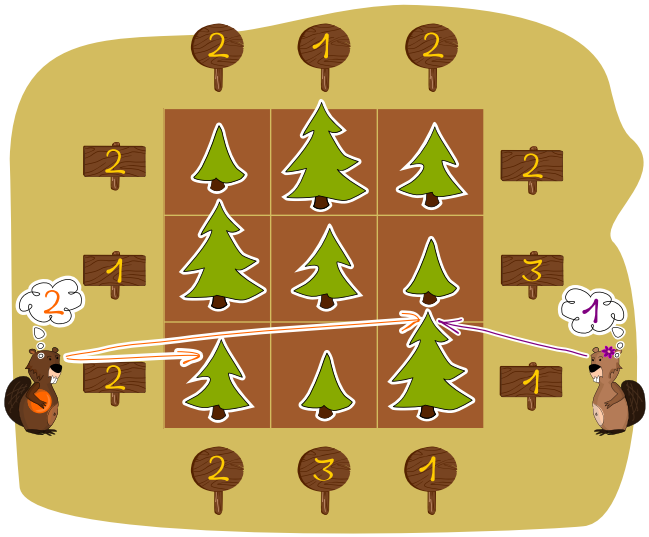
Statt nur neue Zeichen für Ziffern zusammenzustellen, kann man kann auch eigene neue Geheimschriften für Texte erfinden. Dazu schreibt man in eine Tabelle alle Buchstaben und erfindet neue Symbole für die Spalten und Zeilen und erhält so für alle Buchstaben neue Zeichen.

Stichwörter und Webseiten

- Kryptographie (Substitution): `https://de.wikipedia.org/wiki/Kryptographie`
- Geheimschriften

3. 3×3-Tannen-Sudoku

Biber pflanzen Tannen in Reihen. Die Tannen haben drei unterschiedliche Höhen (1, 2 und 3) und in jeder Reihe gibt es genau eine Tanne von jeder Höhe.



Wenn sich die Biber eine Tannenreihe von einem Ende her anschauen, dann können sie niedrigere Tannen, die hinter höheren Tannen versteckt sind, **nicht** sehen.

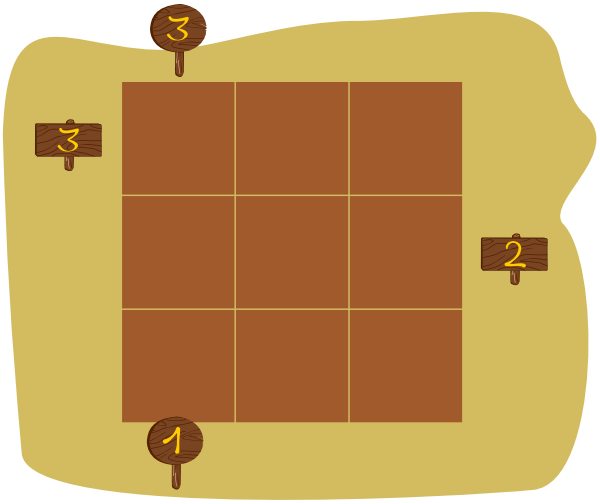
Am Ende jeder Tannenreihe steht auf einem Schild, wie viele Tannen ein Biber von dieser Stelle sehen kann.

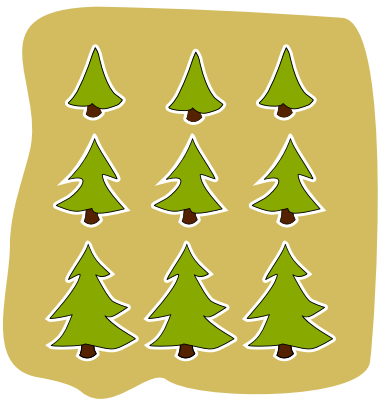
Nun pflanzen die Biber neun Tannen in ein 3×3-Feld, wie im Beispiel rechts.

Dabei gelten folgende Regeln:

- In jeder Zeile (horizontalen Reihe) gibt es genau eine Tanne von jeder Höhe.
- In jeder Spalte (vertikalen Reihe) gibt es genau eine Tanne von jeder Höhe.
- Die Schilder mit der Anzahl sichtbarer Tannen stehen rund um das 3×3-Feld.

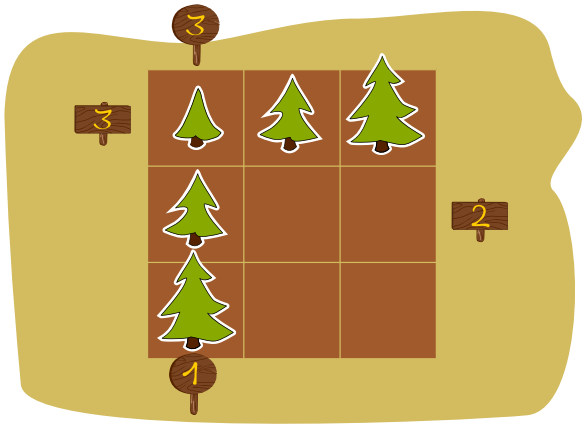
Verteile die Tannen auf die richtigen Felder.





Lösung

Im Feld zeigen zwei Schilder, dass von diesen Positionen drei Tannen gesehen werden können. Alle drei Tannen einer Reihe kann man nur sehen, wenn die Tannen so geordnet sind, dass ihre Höhe ansteigt, also von dieser Position weg. Damit sind die Spalte links und die oberste Zeile bestimmt:

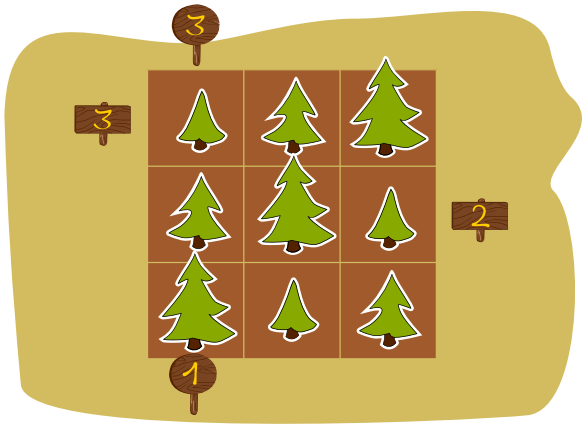


Das Schild rechts mit der 2 verlangt, dass von dort zwei Tannen sichtbar sind, also muss ganz in der Mitte eine Tanne der Höhe 3 sein und diese mittlere Zeile ist somit (), 3 (), 1 ().

Die weiteren Felder werden gemäss der «Sudoku»-Regel gefüllt, dass von jeder Höhe genau eine Tanne in jeder Reihe sein muss.

In der Mitte der untersten Zeile muss eine Tanne der Höhe 1 () stehen, weil für in der mittleren Spalte die beiden anderen Höhen bereits vergeben sind. Ganz rechts unten muss schliesslich eine Tanne der Höhe 2 () folgen, um die Reihe vollständig zu machen.

Die vollständige Lösung sieht so aus:



Dies ist Informatik!

Diese Aufgabe fokussiert auf drei grundlegenden Kompetenzen von Informatikerinnen und Informatikern.

Zuerst geht es darum, eine Lösung zu finden, die gegebene Einschränkungen einhält, oder nach Bedarf einen Lösungsvorschlag zu korrigieren.

Zweitens geht es um die Fähigkeit, Objekte über ihre Darstellung aus Teilinformationen rekonstruieren zu können. Das hängt mit der Generierung von Objekten (Objektdarstellungen) aus eingeschränkten verfügbaren Informationen zusammen, wenn man die Gesetzmässigkeit der Objekte kennt. Solche Vorgehensweisen kann man auch bei der Komprimierung anwenden.

Drittens kann man solche Baumfelder mit Schildern zur Erzeugung von selbst-verifizierenden Codierungen einsetzen. Vorkommende Fehler beim Eintragen oder beim Informationstransport können dann automatisiert erkannt oder sogar korrigiert werden.

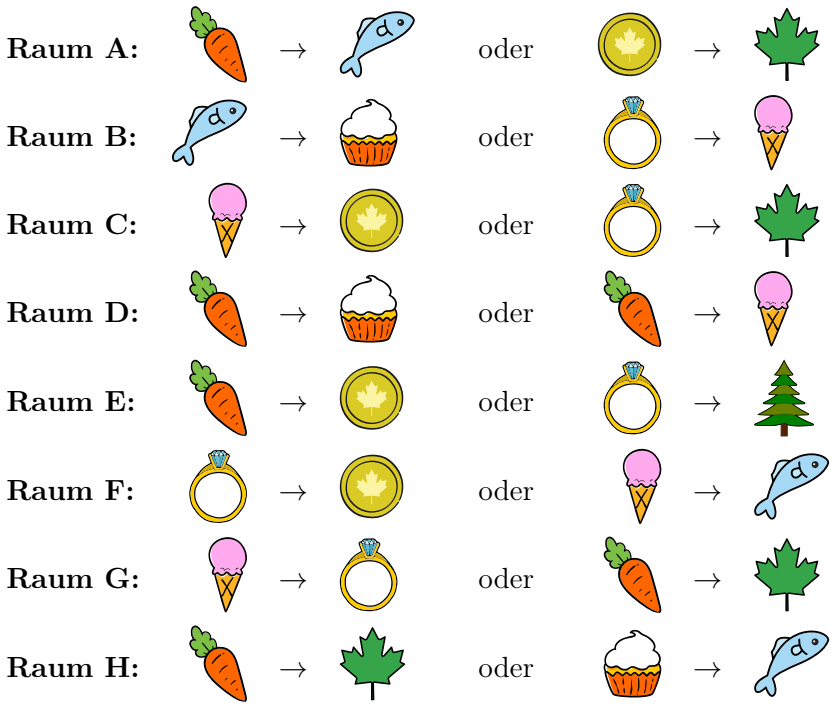
Stichwörter und Webseiten

- Sudoku: `https://de.wikipedia.org/wiki/Sudoku`
- Fehlererkennung und Fehlerkorrektur:
  `https://de.wikipedia.org/wiki/Fehlerkorrekturverfahren`
- Rekonstruktion von Objekten aus Teilinformationen
- Überprüfung der Korrektheit von Datendarstellungen

4. Biber im Schloss

Ein schlauer Biber braucht einen Tannenbaum um im Fluss einen Damm zu bauen. Leider hat er aber nur ein Rüebli . Im Schloss ist heute Markttag und der Biber will dort sein Rüebli gegen einen Tannenbaum eintauschen.

Jeder Raum des Schlosses bietet zwei Tauschangebote. Die Tabelle zeigt diese Angebote:



Zum Beispiel kann der Biber in Raum B für einen Ring ein Cornet bekommen, aber nicht umgekehrt.

In welcher Reihenfolge soll der schlaue Biber durch die Räume gehen, um letztlich den gewünschten Tannenbaum zu besitzen?

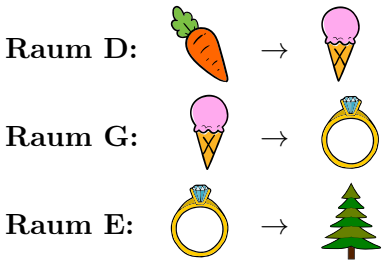
A) DGE: Zuerst Raum D, dann Raum G und zuletzt Raum E.
B) GGE: Zuerst Raum G, dann nochmal Raum G und zuletzt Raum E.
C) AGE: Zuerst Raum A, dann Raum G und zuletzt Raum E.
D) DBC: Zuerst Raum D, dann Raum B und zuletzt Raum C.

Lösung

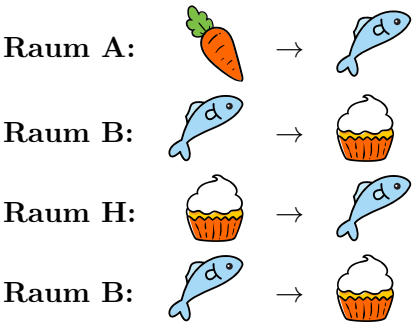
Die richtige Antwort ist A) DGE: Zuerst Raum D, dann Raum G und zuletzt Raum E.

Im Raum D tauscht der Biber sein Rüebli gegen ein Cornet. Danach geht er in Raum G, wo er das Cornet gegen einen Ring tauscht. Am Ende geht der Biber in den Raum E, um den Ring gegen einen Tannenbaum zu tauschen.

Diese Gesamtabfolge sieht so aus:



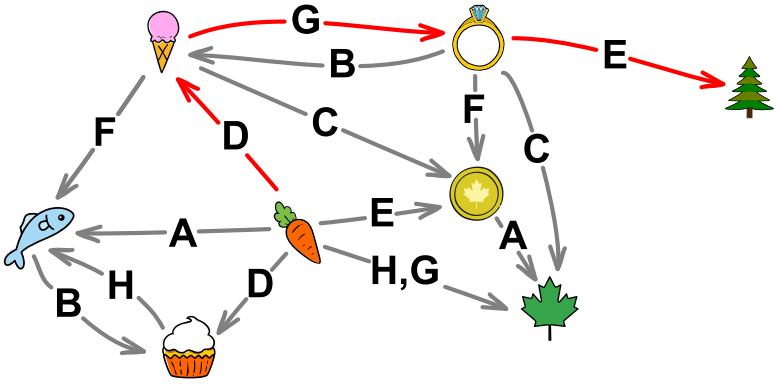
Um eine passende Reihenfolge der Räume zu finden, sind zwei unterschiedliche Strategien zielführend. Die erste Strategie versucht alle Tauschmöglichkeiten in Betracht zu ziehen. Sie beginnt damit, dass man beim ersten Tausch das Rüebli in fünf Räumen (A, D, E, G und H) gegen 6 verschiedene Objekte eintauschen kann. Danach werden für diese 6 Objekte wieder alle Eintauschmöglichkeiten betrachtet. Dies ist aufwendig und kann sogar im Kreis laufen, wie in folgendem Beispiel, wo der Biber beliebig oft die Räume B und H besuchen kann:



Somit ist diese erste Strategie sehr aufwendig und nur mit Glück in kurzer Zeit erfolgreich.

Die zweite Strategie führt in dieser konkreten Aufgabe schnell zum Ziel. Sie basiert darauf, die Suche vom gewünschten Ziel her, also dem Tannenbaum, zu beginnen. Nur im Raum E kann der Biber den gewünschten Tannenbaum bekommen. Den Tannenbaum bekommt man nur im Tausch gegen einen Ring. Das nächste gewünschte Objekt ist also ein Ring! Auch den Ring kann man nur in einem Raum bekommen, nämlich im Raum G im Tausch gegen ein Cornet. Ein Cornet erhält man entweder in Raum B gegen einen Ring oder in Raum D gegen ein Rüebli. Der schlaue Biber muss also seine Tauschvorgänge in Raum D beginnen.

Für eine bessere Übersicht kann die Tabelle der möglichen Tauschvorgänge als Graph mit gerichteten Kanten (Pfeilen) dargestellt werden. Jeder Knoten des Graphen steht für ein Tauschobjekt und jede ausgehende Kante steht für eine Tauschmöglichkeit. Zusätzlich steht auf der Kante, in welchem Raum diese Tauschmöglichkeit besteht.



Diese visuelle Darstellung der Tauschobjekte, Tauschmöglichkeiten und Räume erlaubt es, leicht herauszufinden, wie man vom Rüebli zum Tannenbaum kommt, nämlich auf einem Weg im gerichteten Graphen: Zuerst Raum D, dann Raum G und zuletzt Raum E.

Dies ist Informatik!

Berechnungsprozesse kann man auf einer allgemeinen Ebene betrachten als *Folgen von Transformationen* (hier sind es Tauschvorgänge) oder gleichwertig als *Folgen von Zuständen* eines Systems. Der Startzustand des Systems ist in unserem Fall das Rüebli und die Transformation (der *Berechnungsschritt*) vom Rüebli zum Cornet ändert diesen Zustand zum Cornet.

Ein Berechnungsschritt führt also von einem Zustand zu einem anderen. Eine Abfolge von Berechnungsschritten nennt man auch eine Berechnung.

Somit behandelt diese Aufgabe auch Berechnungen auf einer sehr allgemeinen Ebene. Das System ist im vorliegenden Fall *nichtdeterministisch*; das bedeutet, dass es manchmal mehrere mögliche Berechnungsschritte gibt, also wie in der Aufgabe mehrere Tauschmöglichkeiten. Nichtdeterminismus ist ein weiteres wichtiges Konzept der Modellierung in der Informatik. Die möglichen Berechnungsschritte werden durch *Transformationsregeln* (die Tabelle mit Tauschmöglichkeiten) beschrieben. Zu bestimmen, ob der Biber ein Rüebli in einen Tannenbaum eintauschen kann, also ob ein bestimmter *Zielzustand* des Systems von einem bestimmten *Startzustand* aus erreichbar ist, ist das berühmte *Erreichbarkeitsproblem* mit zahlreichen Anwendungen.

Die Aufgabe oben zeigt, dass es manchmal eine gute Idee ist, vom Zielzustand her den Startzustand zu suchen statt umgekehrt. Diese Strategie nennt man auch *Rückwärtssuche*.

Beim Vergleich der verschiedenen Lösungsstrategien erkennt man, dass der gerichtete Graph eine anschauliche Möglichkeit der Darstellung eines sogenannten *Zustandsraumes* eines Systems mit allen möglichen Übergängen zwischen Zuständen ist. In diesem Basismodell könnte man die bekannten grundlegenden *Suchalgorithmen* in Graphen, nämlich *Breitensuche* und *Tiefensuche* ansprechen.

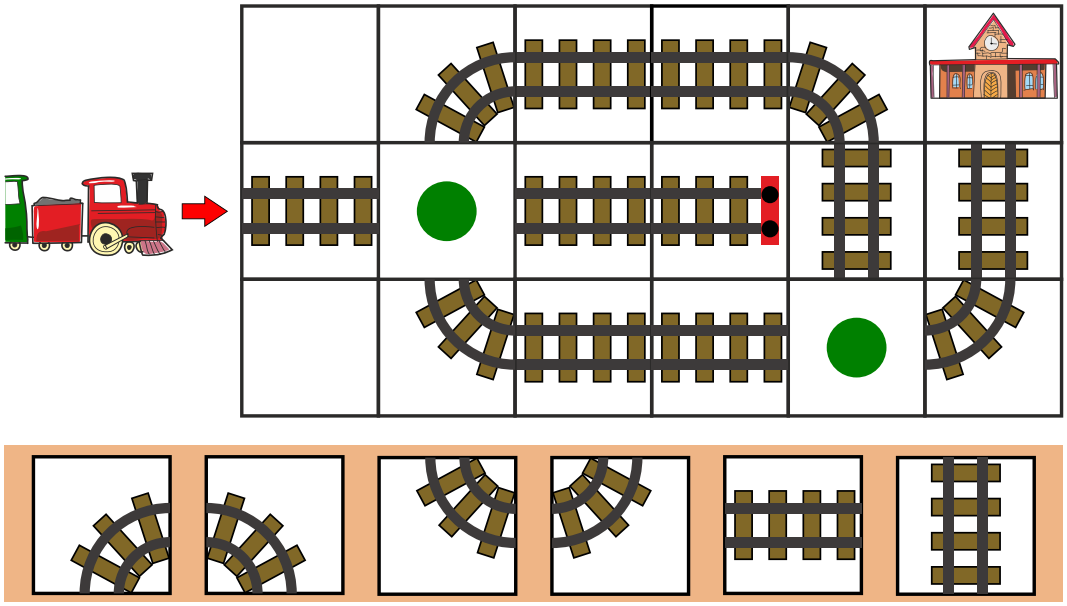
Stichwörter und Webseiten

- Graphentheorie: `https://de.wikipedia.org/wiki/Graph_(Graphentheorie)`

- Erreichbarkeitsproblem:
  `https://de.wikipedia.org/wiki/Erreichbarkeitsproblem_in_Graphen`
- Tiefensuche: `https://de.wikipedia.org/wiki/Tiefensuche`
- Breitensuche: `https://de.wikipedia.org/wiki/Breitensuche`

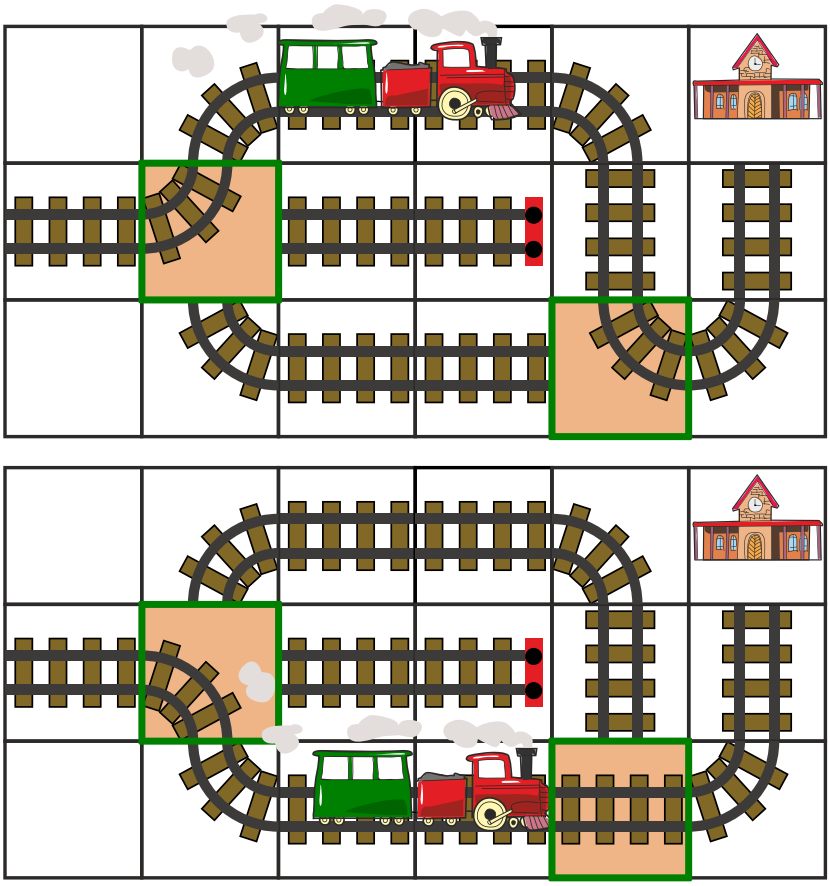
5. Nächster Halt, Bahnhof!

Lege Schienen auf die grünen Punkte, so dass der Zug zum Bahnhof fahren kann.



Lösung

Für dieses Problem gibt es folgende 2 Lösungen:



Bei anderen Kombinationen entgleist der Zug oder fährt gegen den Prellbock.

Dies ist Informatik!

Wie ein Zug stur entlang den Schienenstücken fährt, führt ein Computer stur die Anweisungen eines Programms aus. Er kann nicht erkennen, wenn das Programm einen Fehler enthält und kann dann «abstürzen», so wie ein Zug entgleisen kann, wenn die Schienen falsch gebaut wurden. Beim Schreiben eines Programms muss man also viel sorgfältiger sein, als wenn man beispielsweise einer Person den Weg zum Bahnhof erklärt.

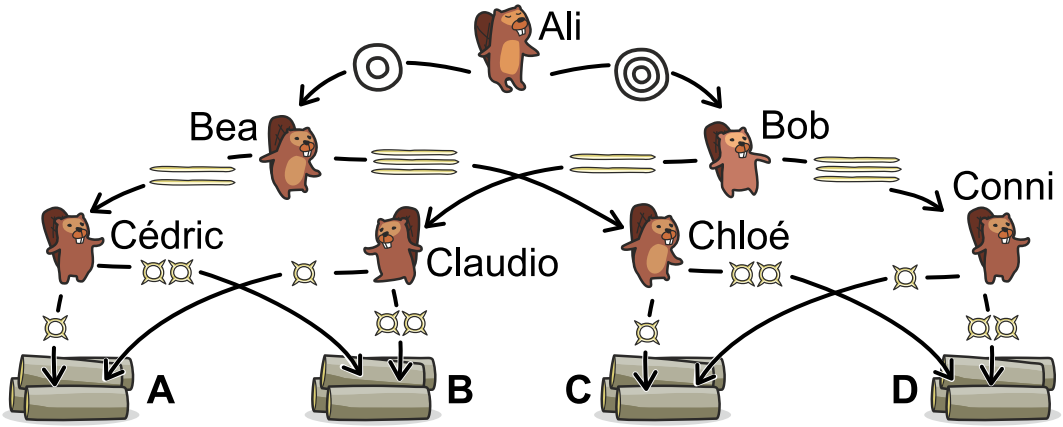
In der Aufgabe oben geht es darum, in einem Programm an den richtigen Stellen fehlende Befehle einzufügen, sodass die Zielsetzung erreicht wird.

Stichwörter und Webseiten

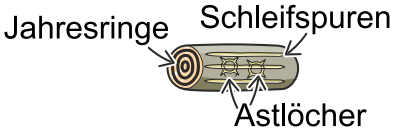
- Programm
- Anweisung: `https://de.wikipedia.org/wiki/Anweisung_(Programmierung)`
- `https://de.wikipedia.org/wiki/Algorithmus`

6. Baumstämme auf Stapel

Im Biberdorf werden die Stämme nach drei Eigenschaften (Anzahl Jahresringe, Anzahl Schleifspuren in der Rinde und Anzahl der Astlöcher) in vier Gruppen (A, B, C, D) verteilt. Wie das abläuft, zeigt das Entscheidungsdiagramm.



Beispielsweise wird dieser Stamm aufgrund folgender Entscheidungen auf den Stapel D gelegt:



- Ali sieht drei Jahresringe und gibt den Stamm an Bob.
- Bob sieht drei Schleifspuren und gibt den Stamm an Conni.
- Conni sieht zwei Astlöcher und legt den Stamm auf den Stapel D.

Auf welchem Stapel wird dieser Stamm abgelegt?

A) Stapel A
B) Stapel B
C) Stapel C
D) Stapel D

Lösung

Die korrekte Antwort ist Stapel C. Dies ist so, weil Ali zwei Jahresringe sieht und den Stamm an Bea gibt. Bea sieht drei Schleifspuren und gibt den Stamm an Chloé weiter. Chloé sieht ein Astloch und legt den Stamm auf den Stapel C.

Wenn man will, kann man für jeden Stapel bestimmen, welche Stämme auf den jeweiligen Stapel gehören. Auf jedem Stapel gibt zwei Arten von Stämmen.

Auf Stapel **A**:

- Stämme mit 2 Jahresringen, 2 Schleifspuren und 1 Astloch.
- Stämme mit 3 Jahresringen, 2 Schleifspuren und 1 Astloch.

Auf Stapel **B**:

- Stämme mit 2 Jahresringen, 2 Schleifspuren und 2 Astlöchern
- Stämme mit 3 Jahresringen, 2 Schleifspuren und 2 Astlöchern

Auf Stapel **C**:

- Stämme mit 2 Jahresringen, 3 Schleifspuren und 1 Astloch
- Stämme mit 3 Jahresringen, 3 Schleifspuren und 1 Astloch

Auf Stapel **D**:

- Stämme mit 2 Jahresringen, 3 Schleifspuren und 2 Astlöchern
- Stämme mit 3 Jahresringen, 3 Schleifspuren und 2 Astlöchern

Dies ist Informatik!

Diese Aufgabe berührt mehrere Konzepte der Informatik.

Vor allem wird das Konzept der *Entscheidungsdiagramme* angesprochen, die sehr vielseitige Anwendungen in der Informatik haben. Hier verwendet man sie zur *Klassifizierung* von Objekten in gewählte Kategorien. (Sehr häufig sind es *Entscheidungsbäume*, eine spezielle Art von Entscheidungsdiagrammen. Das Entscheidungsdiagramm der Aufgabe ist hier kein Entscheidungsbaum, weil auf der untersten Ebene je zwei Gruppen auf denselben Stapel gelegt werden.)

Man kann das Entscheidungsdiagramm hier auch als eine abstrakte Darstellung der Werte einer Funktion von mehreren Variablen ansehen. Terminologisch genau spricht man in der Informatik von «branching programs».

Zudem spricht man hier auch das Konzept der *Attribute* (Merkmale oder Eigenschaften) von Objekten an. Hier haben die Objekte drei Attribute (Jahresringe, Schleifspuren, Astlöcher), wobei jedes Attribut zwei mögliche Werte hat (zwei oder drei Jahresringe oder Schleifspuren und ein oder zwei Astlöcher).

Es gibt viele Anwendungsmöglichkeiten für solche Entscheidungsdiagramme. Eine davon ist die Klassifizierung von Paketen beim Versenden durch ein Netzwerk (mit Routern oder Switches).

Stichwörter und Webseiten

- Entscheidungsbaum: `https://de.wikipedia.org/wiki/Entscheidungsbaum`
- Klassifizierung

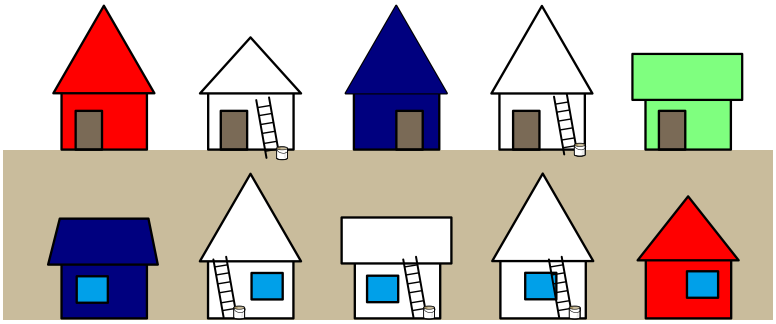
7. Farbiges Quartier

Die Anwohner einer Strasse wollen ihre weissen Häuser farbig anmalen. Jedes Haus soll eine von drei Farben bekommen: Hellgrün, Rot oder Dunkelblau. Damit es nicht langweilig aussieht, gelten folgende Regeln:

- Zwei direkt nebeneinander stehende Häuser dürfen nicht dieselbe Farbe haben.
- Zwei Häuser, die sich auf der Strasse direkt gegenüber stehen, dürfen nicht dieselbe Farbe haben.

Einige Anwohner haben ihre Häuser bereits farbig angemalt. Die restlichen Anwohner müssen jetzt ihre Häuser so anmalen, dass die Regeln nicht verletzt werden.

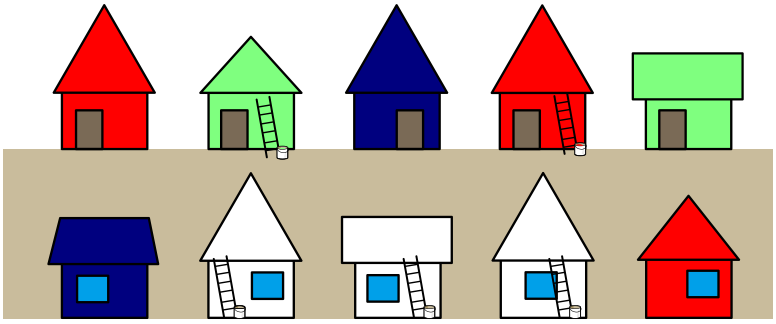
Finde für die Anwohner passende Farben.



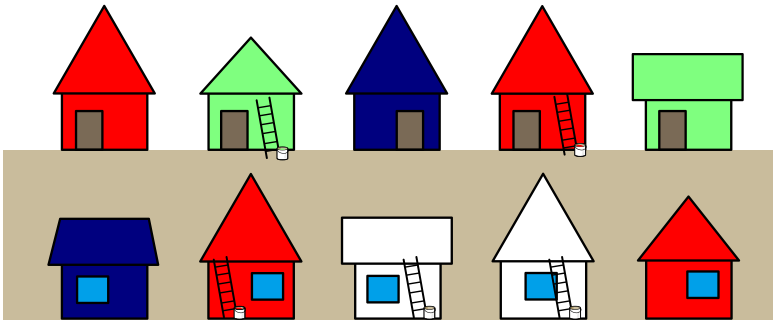
Lösung

Die Farben der Häuser lassen sich am einfachsten schrittweise herausfinden.

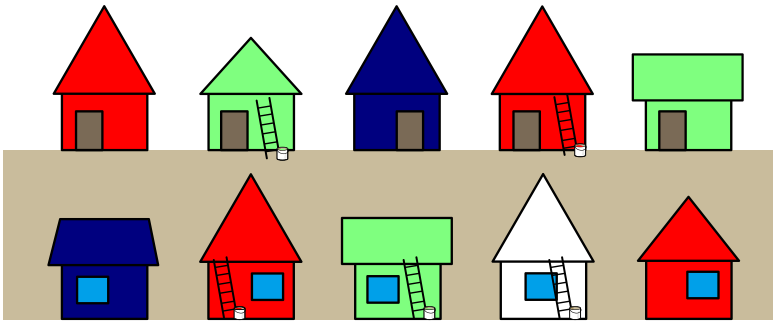
Die beiden weissen Häuser auf der oberen Strassenseite sind jeweils von zwei verschiedenfarbigen Häusern links und rechts umgeben. Deshalb können sie jeweils nur in einer ganz bestimmten Farbe angemalt werden, ohne die Regeln zu verletzen: das weisse Haus links oben in Hellgrün und das rechte weisse Haus oben in Rot.



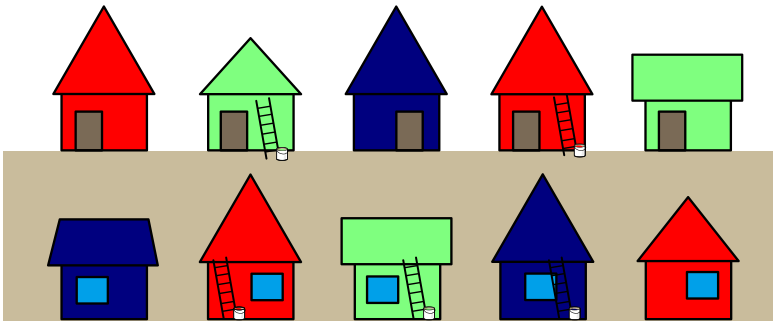
Als Nächstes kann man feststellen, dass das weisse Haus links unten rot angemalt werden muss, weil das Haus direkt links daneben dunkelblau und das Haus direkt gegenüber hellgrün ist:



Fast dieselbe Überlegung kann man jetzt für das mittlere Haus der unteren Strassenseite machen: Es muss hellgrün angemalt werden, weil direkt links davon das gerade eben rot angemalte Haus steht und gegenüber ein dunkelblaues Haus.



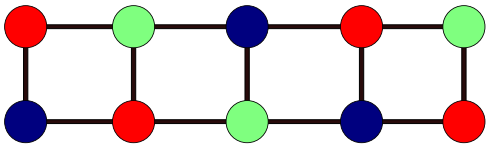
Zuletzt kann man auch für das rechte weisse Haus in der unteren Strassenseite die Farbe bestimmen: Das Haus direkt rechts daneben und das Haus direkt gegenüber sind zwar beide rot, da aber das Haus direkt links daneben jetzt hellgrün ist, bleibt nur noch die Möglichkeit, das Haus dunkelblau anzumalen:



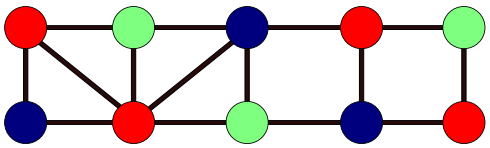
Dies ist Informatik!

Abstrakt gesehen, geht es in dieser Aufgabe darum, eine Lösung zu finden, die die vorgegebenen Einschränkungen (Regeln) erfüllt. Dies ist in der Informatik eine sehr häufige Problemstellung.

Die Häuser und deren direkte Nachbarschaften (sowohl nach links und nach rechts als auch quer über die Strasse hinweg) können gut mit Hilfe eines *Graphen* modelliert werden, einer in der Informatik weit verbreiteten Datenstruktur. Dabei ist jedes Haus ein *Knoten* und jede direkte Nachbarschaft eine *Kante*:



Im Bild sind die Knoten bereits so gefärbt wie die entsprechenden Häuser. Für die Häuser gab es die Regel, dass benachbarte Häuser unterschiedliche Farben haben müssen. Im Bild ist die Färbung der Knoten deshalb so, dass direkt über eine Kante verbundene Knoten nie dieselbe Farbe haben. Dass es eine solche *gültige Färbung* des Graphen mit drei Farben gibt, ist nicht selbstverständlich. Wenn man zwei Kanten so ergänzt, wie im nächsten Bild, dann gibt es keine gültige Färbung mehr: Egal wie man in diesem Graphen die drei Farben verteilt, es gibt immer zwei direkt verbundene Knoten mit derselben Farbe.



Mit vier Farben geht es aber wieder. Vielleicht geht es immer mit vier Farben? Die Antwort ist wieder nein. Doch zumindest eine bestimmte Art von Graphen kann man immer mit vier Farben gültig einfärben: nämlich sogenannte *planare Graphen*. Das sind Graphen, die man so zeichnen kann, dass sich keine Kanten überkreuzen. (Der Graph im letzten Bild ist nicht planar, nämlich wegen

den Verbindungen der vier Knoten ganz links.) Dass planare Graphen eine gültige Färbung mit vier Farben haben, nennt man den *Vier-Farben-Satz*.

Besonders interessant ist der Vier-Farben-Satz für das Erstellen von Landkarten. Wenn man sich jedes Land als Knoten vorstellt und dann benachbarte Länder mit einer Kante verbindet, erhält man immer einen planaren Graphen. (Genau genommen müssen wir dafür die Existenz von sogenannten Enklaven und Exklaven ausschliessen, also Teile von einem Land, die komplett in einem anderen Land liegen.) Diesen Graphen kann man also mit vier Farben gültig einfärben, und deshalb kann man auch die entsprechenden Länder auf der Karte mit vier Farben so einfärben, dass benachbarte Länder immer verschiedene Farben haben.

Der Beweis, dass vier Farben genügen, ist nicht einfach. Dass fünf Farben genügen, wusste man bereits vor 200 Jahren. Dass vier Farben genügen, haben die Mathematiker Kenneth Appel und Wolfgang Haken dann im Jahr 1976 bewiesen. Dabei haben einen Computer verwendet, um eine Vielzahl von Ausnahmen und Gegenbeispielen zu überprüfen. Der Computer hat über tausend Stunden dafür gebraucht. Alles von Hand zu prüfen, wäre völlig unmöglich gewesen. Viele Mathematiker stellten dann die Frage, ob ein solcher Beweis überhaupt gültig ist, weil man dem Computer vertrauen muss.

Stichwörter und Webseiten

- Dreifarbenproblem: `https://de.wikipedia.org/wiki/Dreifarbenproblem`
- Vier-Farben-Satz: `https://de.wikipedia.org/wiki/Vier-Farben-Satz`
- `https://de.wikipedia.org/wiki/Enklave`

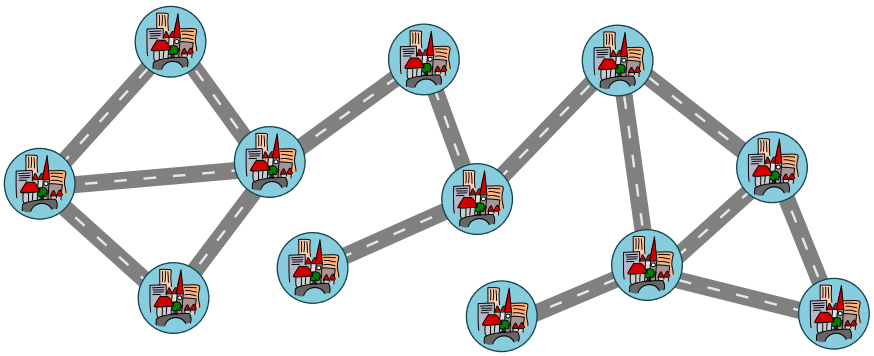
8. Epidemische Überlegungen

Biberland besteht aus 12 Städten, die durch Strassen miteinander verbunden sind. Städte, die direkt oder indirekt durch Strassen miteinander verbunden sind, bilden eine Handelsgemeinschaft. Die Karte zeigt also in der aktuellen Form eine einzelne Handelsgemeinschaft aus 12 Städten.

Um eine Epidemie einzudämmen, soll der Verkehr reduziert werden. Das Biberparlament beschliesst, genau zwei Strassen zu sperren, um die Städte in drei einzelne Handelsgemeinschaften aufzuteilen.

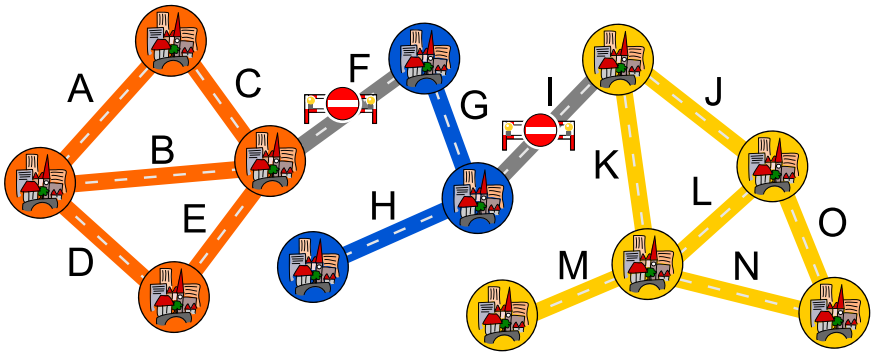
Um niemanden mehr als notwendig zu isolieren, soll die kleinste Handelsgemeinschaft aus möglichst vielen Städten bestehen.

Welche zwei Strassen sollen gesperrt werden?



Lösung

Die richtige Antwort ist: Die Strassen F und I im Bild unten werden gesperrt. So entsteht je eine Handelsgemeinschaft aus 3, 4 und 5 Städten.



Es liegt auf der Hand, dass wir nur Strassen betrachten müssen, die bei einer Sperrung auch die Teilung der Handelsgemeinschaft bewirken, weil sie die einzige Verbindung darstellen. Denn wir brauchen ja zwei echte Teilungen, um zu drei Einheiten zu gelangen. So bringt es zum Beispiel nichts, Strasse B zu sperren, weil man über A und C immer noch alle Städte erreichen kann. Es bleiben daher für die Sperrung nur die Kandidaten F, G, H, I und M übrig.

Probiert man alle 10 Möglichkeiten durch, zwei der fünf Strassen zu sperren, kommt man auf obige Antwort. Als Mensch sieht man zudem sofort, dass die Sperrung von H oder M nur eine einzelne Stadt abtrennen würde und daher nicht in Frage kommt. Das schränkt die Zahl der zu betrachtenden Möglichkeiten weiter ein.

Dies ist Informatik!

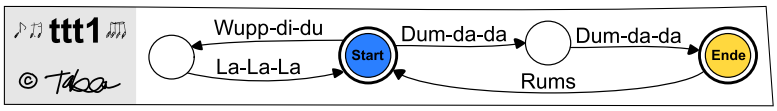
In der Informatik will man ein gegebenes Netzwerk häufig in sogenannte *Zusammenhangskomponenten* aufteilen. In einer Zusammenhangskomponente sind alle Teile mit über direkte oder indirekte Wege miteinander verbunden, während es zwischen verschiedenen Zusammenhangskomponenten überhaupt keine Verbindung gibt. Auf der Hand liegt die Anwendung bei Computernetzen, in denen relevant ist, welche Computer von welchen anderen erreicht werden können. Aber auch zum Beispiel bei der Schrifterkennung (OCR) ist es eine wichtige Information, welche Punkte «verbunden» sind.

Stichwörter und Webseiten

- Zusammenhangkomponenten:
  `https://de.wikipedia.org/wiki/Zusammenhang_(Graphentheorie)`
- Traversierung von Graphen

9. Tabeas taktvolle Texte

Tabea ist sehr erfolgreich mit ihren Liedtexten der Marke ttt: Tabeas Taktvolle Texte. Diese können mit dem folgenden Diagramm ttt1 produziert werden:



Um ein Lied zu produzieren, beginnt Tabea bei «Start» 

und folgt einem der ausgehenden Pfeile. Bei mehreren Möglichkeiten darf sie einen aussuchen. Sie singt die entsprechenden Silben auf dem Weg in der gegebenen Reihenfolge. Erreicht sie «Ende» 

, darf das Lied zu Ende sein, kann aber auch weitergehen.

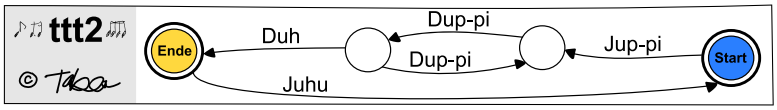
Mögliche Lieder sind zum Beispiel:

«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»

oder

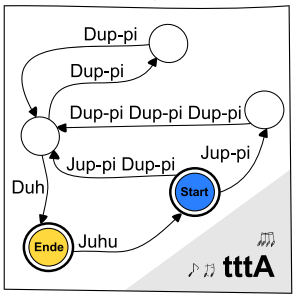
«Dum-da-da Dum-da-da Rums Wupp-di-du La-La-La
Dum-da-da Dum-da-da Rums Wupp-di-du La-La-La
Dum-da-da Dum-da-da Rums Dum-da-da Dum-da-da»

Tabea geht im November 2020 mit neuen Texten nach ttt2 in Produktion:

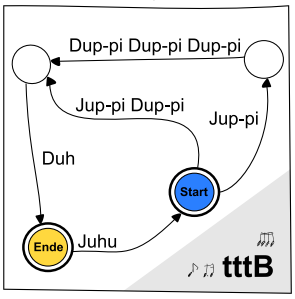


Mit welchem der folgenden Diagramme können genau dieselben Liedtexte wie mit ttt2 produziert werden?

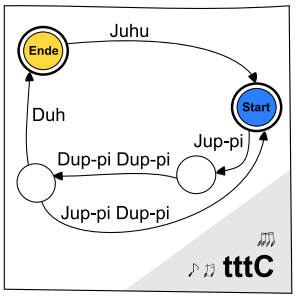
A)



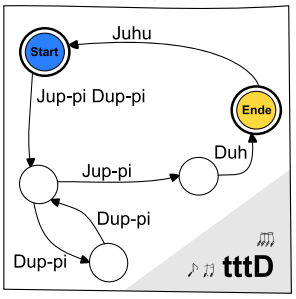
B)



C)

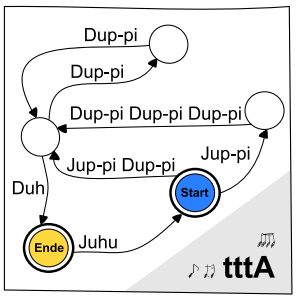


D)

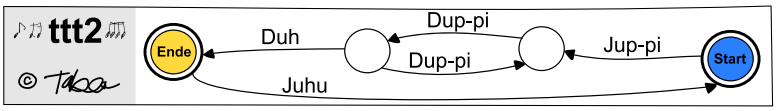


Lösung

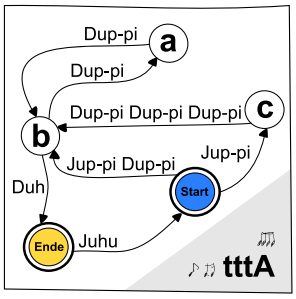
Die korrekte Antwort ist A) Diagramm tttA:



Produziert man ein Lied mit Diagramm ttt2, startet es in jedem Fall mit «Jup-pi» und es folgt mindestens ein «Dup-pi». Jetzt geht es entweder direkt mit «Duh» weiter oder einer geraden Anzahl weiterer «Dup-pi» und danach «Duh». Nun kann das Lied zu Ende sein oder mit einem «Juhu» fortfahren und wieder von vorne anfangen.



Das Diagramm tttA erreicht genau das Gleiche: Vom «Start» aus kann das Lied entweder direkt zu **b** und so mit «Jup-pi Dup-pi» beginnen oder über **c** mit «Jup-pi Dup-pi Dup-pi Dup-pi». Danach folgt mit einem Umweg über **a** alternativ noch eine beliebige gerade Zahl an «Dup-pi», dann kommt man mit «Duh» zum Ende des Liedes. Genau wie in ttt2 kann man nach «Juhu» wieder von Neuem beginnen.



Sowohl ttt2 als auch tttA können nach dem anfänglichen «Jup-pi» eine beliebige ungerade Anzahl von ununterbrochen aufeinanderfolgenden «Dup-pi» erzeugen. Im Gegensatz dazu kann tttB nur 1 oder 3 ununterbrochen aufeinanderfolgende «Dup-pi» erzeugen und tttC nur 1 oder 2. Und tttD kann zwar eine ungerade Anzahl von ununterbrochen aufeinanderfolgenden «Dup-pi» erzeugen, stellt aber dem abschliessenden «Duh» immer ein zusätzliches «Jup-pi» voran, das ttt2 dort nicht erzeugen kann.

Daher ist tttA die einzige mögliche Antwort.

Dies ist Informatik!

Eine wichtige Aufgabe der Informatik ist es, Strukturen in Daten zu erkennen, zum Beispiel in Sprache wie etwa einem Liedtext. Die Diagramme repräsentieren sogenannte *Endliche Automaten*, mit denen sehr strikte Regeln für das Erzeugen und Erkennen bestimmter Sprachen definiert werden können. Das ist wiederum entscheidend bei der Entwicklung von Programmiersprachen. In unserem Beispiel beschreibt der Endliche Automat die Menge von Liedern, die mit diesem erzeugt werden können.

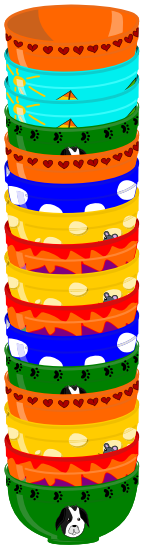
Mustererkennung ist aber auch in vielen anderen Bereichen wichtig, etwa der Verarbeitung natürlicher Sprache.

Stichwörter und Webseiten

- Endliche Automaten:
  `https://de.wikipedia.org/wiki/Deterministischer_endlicher_Automat`
- Formale Sprachen: `https://de.wikipedia.org/wiki/Formale_Sprache`
- `https://sites.google.com/isabc.ca/computationalthinking/pattern-recognition`

10. Schälchen-Stapel

Drei Geschwister wollen beim Morgenessen aus drei gleichen Schälchen essen. Sie haben einen hohen Stapel von Schälchen. Vorsichtshalber nehmen sie immer nur einzelne Schälchen vom oberen Ende des Stapels.

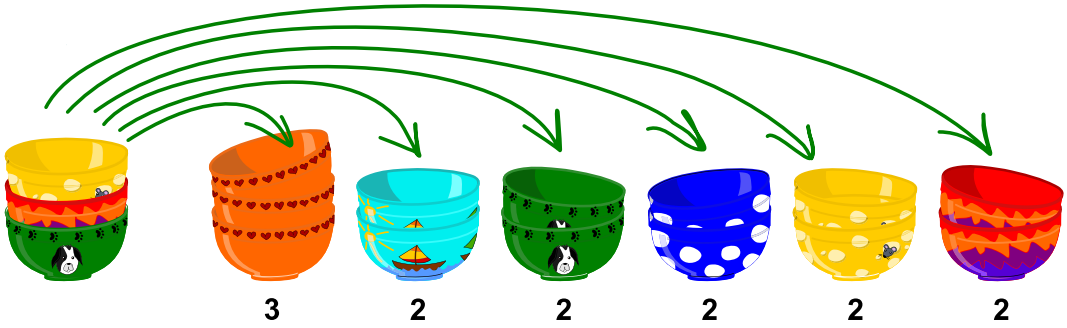


Was ist die kleinste Anzahl von Schälchen, die sie vom abgebildeten Stapel nehmen müssen, um drei gleiche zu haben?

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

Lösung

Antwort K): Mindestens 13 Schälchen müssen vom Stapel genommen werden, um drei gleichartige Schälchen zu erhalten.



Dies ist Informatik!

Ein *Stapel*, in der Informatik oft *Stack* und manchmal auch *LIFO-Speicher*, *Keller* oder *Kellerspeicher* genannt, ist eine sehr verbreitete Art Dinge zu speichern. Ein Stapel ist eine sehr einfache, aber trotzdem mächtige Struktur, die man beim Programmieren häufig verwendet. Es gibt Regeln, wie man Dinge auf einen Stapel legen kann und wie man sie wieder entnimmt, meistens nämlich nur von oben. In dieser Aufgabe haben wir es nur mit dem Wegnehmen vom Stapel zu tun. Die Regel besagt, dass immer nur das oberste Objekt vom Stapel genommen werden kann. Will man das zehnte Schälchen im Stapel bekommen, muss man also zehn Schälchen einzeln herunternehmen. Dabei ist es wichtig, einen Platz zu haben, wohin man die anderen neun Schälchen stellen kann; das ist beim Programmieren auch so. Haben wir einen zweiten Stapel und können wir die Stapel so hoch machen, wie wir wollen, dann könnten wir damit theoretisch schon alles berechnen, was man mit einem Computer berechnen könnte! (In der Informatik nennt man diese Eigenschaft auch *Turing-Mächtigkeit*.) So einfache Stapel sind wirklich mächtig!

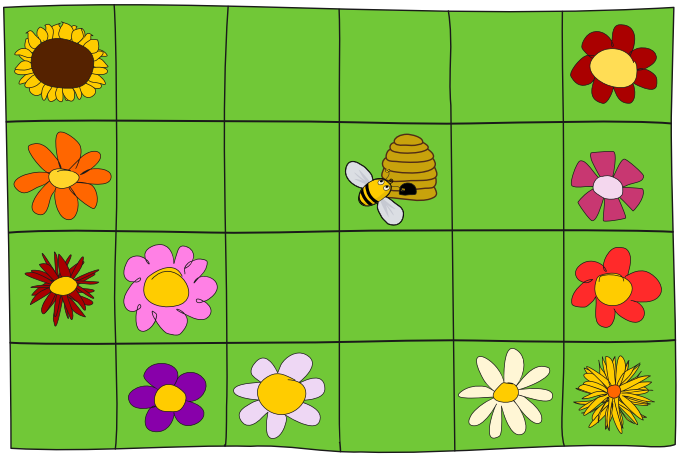
Stichwörter und Webseiten

- Stapelspeicher: `https://de.wikipedia.org/wiki/Stapelspeicher`
- Turingmaschine: `https://de.wikipedia.org/wiki/Turingmaschine`

11. Summ, summ, summ...

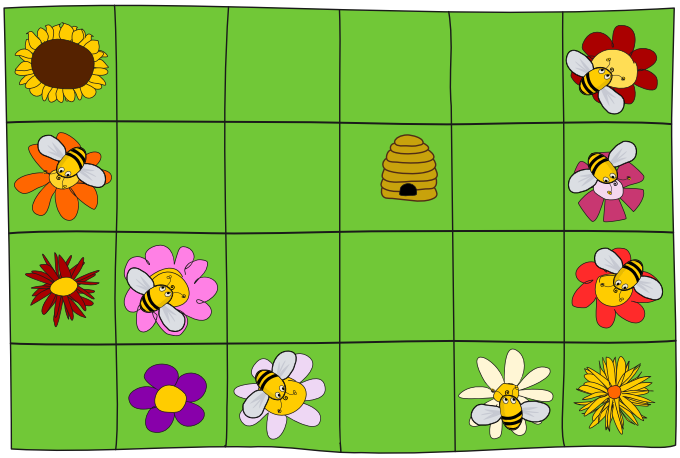
Eine Biene fliegt in 10 Minuten ein Feld nach oben, unten, links oder rechts. Sie fliegt vom Bienenstock aus höchstens 30 Minuten, bevor sie umkehrt.

Welche Blumen sind vom Bienenstock aus in höchstens 30 Minuten erreichbar?



Lösung

Die Blumen mit Biene darauf sind vom Bienenstock aus in höchstens 30 Minuten erreichbar:



Das Bild unten zeigt für jedes Feld, wie viele Minuten eine Biene braucht, um es vom Bienenstock aus zu erreichen. In einer halben Stunde sind also alle Felder erreichbar, in denen 10, 20 oder 30 steht.

Das Ausfüllen mit den Zahlen geht so: In die Felder neben dem Bienenstock schreiben wir 10 drin, weil die Biene 10 Minuten braucht, um vom Bienenstock dorthin zu fliegen. Dann schreiben wir 20 in alle leeren Felder neben einem Feld mit 10, weil die Biene in 10 Minuten von einem Feld zum nächsten kommt. Wir machen jetzt immer so weiter. Wir schreiben also 30 in alle leeren Felder neben einem Feld mit 20 drin. Dann schreiben wir 40 in alle leeren Felder neben einem Feld mit 30 drin. Schliesslich schreiben wir 50 in alle leeren Felder neben einem Feld mit 40 drin.

Dies ist Informatik!

Beim Lösen der Aufgabe haben wir für jedes Feld berechnet, in welcher Zeit eine Biene es vom Bienenstock aus erreichen kann. Zuerst werden die in 10 Minuten erreichbaren Felder bestimmt. Diese werden dann benützt, um die 20 Minuten entfernten Felder zu bestimmen. Mit Hilfe der 20 Minuten entfernten Felder werden dann die 30 Minuten entfernten Felder gefunden und so weiter.

Wir verwenden also bereits berechnete und gespeicherte Ergebnisse (die Zahlen der ausgefüllten Felder) zum Berechnen von weiteren Ergebnissen (die Zahlen der benachbarten, noch leeren Felder). Dieses sehr allgemeine Prinzip nennt man *dynamisches Programmieren*. Dabei ist es meist wichtig, in welcher Reihenfolge die Ergebnisse berechnet werden. Dies ist auch beim Bienenflug zu beachten.

In der Aufgabe fliegt eine Biene in 10 Minuten nur nach oben, unten, links oder rechts. Das ist etwas ungewöhnlich, denn in Realität fliegt eine Biene wahrscheinlich auch gerne schräg über die Felder. Mit dieser realistischeren Annahme wären die in einer halben Stunde erreichbaren Felder durch einen Kreis begrenzt anstatt durch eine Raute wie in der Aufgabe.

Die übliche Distanzmessung, die zu einem Kreis führt, heisst *Euklidische Distanz*. Die Distanzmessung aus der Aufgabe, bei der man sich nur so horizontal oder vertikal durch Quadrate bewegen darf, heisst *Manhattan-Metrik*. (Der Name kommt von den gitterförmigen Strassennetzen in modernen Städten wie Manhattan.)

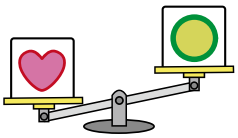
Stichwörter und Webseiten

- Dynamisches Programmieren:
  `https://de.wikipedia.org/wiki/Dynamische_Programmierung`
- Euklidische Distanz: `https://de.wikipedia.org/wiki/Euklidischer_Abstand`
- Manhattan-Distanz: `https://de.wikipedia.org/wiki/Manhattan-Metrik`

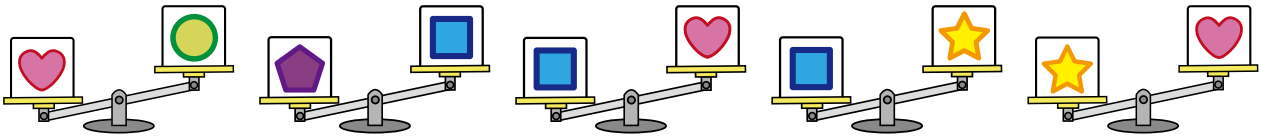
12. Schwere Vergleiche

Fünf Kisten sind mit fünf unterschiedlichen Symbolen gekennzeichnet:  ,  ,  ,  und  .

Mit Hilfe einer Waage werden jeweils zwei Kisten verglichen. Der folgende Vergleich ergibt beispielsweise, dass  schwerer als  ist:



Es werden insgesamt fünf Vergleiche gemacht:



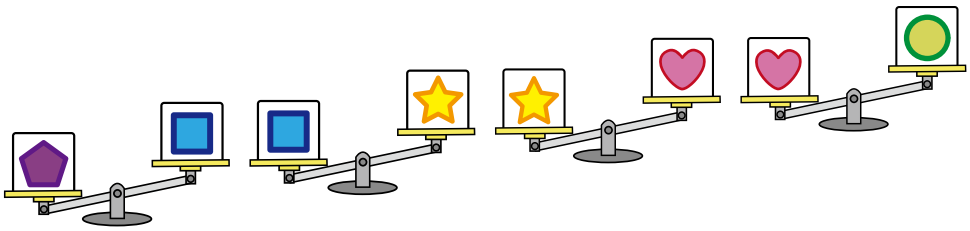
Welche Kiste ist am schwersten?

A) 
B) 
C) 
D) 
E) 

Lösung

Die Kiste C) mit dem Pentagon ist am schwersten.

Die folgende Abbildung enthält vier der fünf gemachten Vergleiche und alle fünf Kisten.



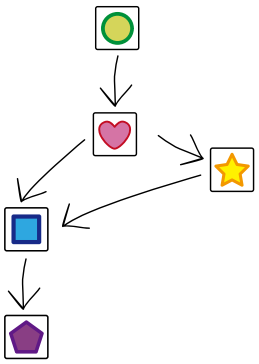
Damit sieht man sofort: Die Kiste mit dem Pentagon ist schwerer als die Kiste mit dem Quadrat . Die Kiste mit dem Quadrat ist schwerer als die Kiste mit dem Stern . Die Kiste mit dem Stern ist schwerer als die Kiste mit dem Herz . Und die Kiste mit dem Herz ist schwerer als die Kiste mit dem Kreis .

Daraus kann man jetzt schliessen, dass die Kiste mit dem Pentagon schwerer ist als alle anderen. Dies liegt an einer speziellen Eigenschaft des Vergleichens von Gewichten: Wenn A schwerer ist als B und B schwerer als C, dann ist auch A schwerer als C. Diese sehr einleuchtende Eigenschaft nennt man *Transitivität*.

Übrigens gibt es einen cleveren Weg, diese Aufgabe abkürzen. Da nach der einen schwersten Kiste gesucht wird, reicht es, einfach nach der Kiste suchen, die keinmal leichter als eine andere Kiste ist, und das ist nur die Kiste mit dem Pentagon .

Dies ist Informatik!

Letztlich geht es in dieser Aufgabe darum, irgendwelche Objekte zu sortieren. Zum Sortieren benützt man in der Informatik häufig spezielle *Graphen*, die aus *Knoten* (den zu sortierenden Objekten) und *Kanten* (Vergleichen zwischen zwei Objekten) bestehen. Die Objekte sind in dieser Aufgabe die Kisten und die Vergleiche sind die Wägungen. Wenn man die Kanten als Pfeile zeichnet, die auf das schwerere Objekt zeigen, dann sieht der Graph für diese Aufgabe so aus:



Die Objekte sollen jetzt so in einer Reihe angeordnet werden, dass die Pfeile immer nur von den Objekten weiter links zu Objekten weiter rechts gehen. Eine solche Anordnung nennt man dann

eine *topologische Sortierung*. Eine topologische Sortierung erhält man sehr einfach, indem man immer wieder einen Knoten aus dem Graphen herausnimmt, auf den kein Pfeil zeigt, und die herausgenommenen Knoten in dieser Reihenfolge hintereinander stellt.

Aber Achtung: Nicht zu jedem Graphen gibt es eine topologische Sortierung. Zum Beispiel existiert keine, wenn es irgendwo drei Kanten gibt, die im Kreis zeigen.

Stichwörter und Webseiten

- Transitivität: `https://de.wikipedia.org/wiki/Transitive_Relation`
- Graph: `https://de.wikipedia.org/wiki/Graph_(Graphentheorie)`
- Topologische Sortierung: `https://de.wikipedia.org/wiki/Topologische_Sortierung`

A. Aufgabenautoren

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
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: Wettbewerb 2020

http://www.haslerstiftung.ch/
Stiftungszweck der Hasler Stiftung ist die Förderung der Informations- und Kommunikationstechnologie (IKT) zum Wohl und Nutzen des Denk- und Werkplatzes Schweiz. Die Stiftung will aktiv dazu beitragen, dass die Schweiz in Wissenschaft und Technologie auch in Zukunft eine führende Stellung innehat.

http://www.baerli-biber.ch/
Schon in der vierten Generation stellt die Familie Bischofberger ihre Appenzeller Köstlichkeiten her. Und die Devise der Bischofbergers ist dabei stets dieselbe geblieben: «Hausgemacht schmeckt's am besten». Es werden nur hochwertige Rohstoffe verwendet: reiner Bienenhonig und Mandeln allererster Güte. Darum ist der Informatik-Biber ein «echtes Biberli».

http://www.verkehrshaus.ch/

Standortförderung beim Amt für Wirtschaft und Arbeit Kanton Zürich

i-factory (Verkehrshaus Luzern)
Die i-factory bietet ein anschauliches und interaktives Erproben von vier Grundtechniken der Informatik und ermöglicht damit einen Erstkontakt mit Informatik als Kulturtechnik. Im optischen Zentrum der i-factory stehen Anwendungsbeispiele zur Informatik aus dem Alltag und insbesondere aus der Verkehrswelt in Form von authentischen Bildern, Filmbeiträgen und Computer-Animationen. Diese Beispiele schlagen die Brücke zwischen der spielerischen Auseinandersetzung in der i-factory und der realen Welt.

http://www.ubs.com/
Wealth Management IT and UBS Switzerland IT

http://www.oxocard.ch/
OXOcard: Spielend programmieren lernen
OXON

https://educatec.ch/
educaTEC
Wir sind MINT-Experten. Seit unserer Gründung 2004 verfolgen wir das Ziel, Technik und ingenieurwissenschaftliches Denken in öffentlichen und privaten Schulen der Schweiz zu fördern. In Kombination mit kompetenter Beratung und Unterstützung offerieren wir Lehrkräften innovative Lehrmaterialien von weltweit führenden Herstellern sowie Lernkonzepte für den MINT-Bereich und verwandte Fächer.

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

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

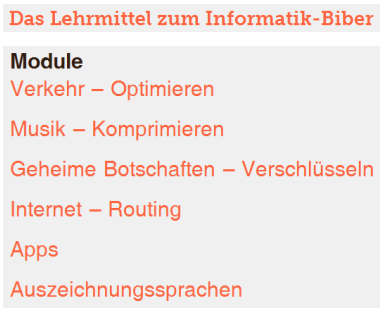
http://www.phlu.ch/
Pädagogische Hochschule Luzern

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. Weiterführende Angebote



`http://informatik-biber.ch/einleitung/`
Das Lehrmittel zum Biber-Wettbewerb ist ein vom SVIA, dem schweizerischen Verein für Informatik in der Ausbildung, initiiertes Projekt und hat die Förderung der Informatik in der Sekundarstufe I zum Ziel.
Das Lehrmittel bringt Jugendlichen auf niederschwellige Weise Konzepte der Informatik näher und zeigt dadurch auf, dass die Informatikbranche vielseitige und spannende Berufsperspektiven bietet.
Lehrpersonen der Sekundarstufe I und weiteren interessierten Lehrkräften steht das Lehrmittel als Ressource zur Vor- und Nachbereitung des Wettbewerbs kostenlos zur Verfügung.
Die sechs Unterrichtseinheiten des Lehrmittels wurden seit Juni 2012 von der LerNetz AG in Zusammenarbeit mit dem Fachdidaktiker und Dozenten Dr. Martin Guggisberg der PH FHNW entwickelt. Das Angebot wurde zweisprachig (Deutsch und Französisch) entwickelt.

I learn it: `http://ilearnit.ch/`
In thematischen Modulen können Kinder und Jugendliche auf dieser Website einen Aspekt der Informatik auf deutsch und französisch selbständig entdecken und damit experimentieren. Derzeit sind sechs Module verfügbar.

Werden Sie SVIA Mitglied – `http://svia-ssie-ssii.ch/svia/mitgliedschaft` und unterstützten Sie damit den Informatik-Biber.
Ordentliches Mitglied des SVIA kann werden, wer an einer schweizerischen Primarschule, Sekundarschule, Mittelschule, Berufsschule, Hochschule oder in der übrigen beruflichen Aus- und Weiterbildung unterrichtet.
Als Kollektivmitglieder können Schulen, Vereine oder andere Organisationen aufgenommen werden.