

Διαγωνισμός BEBRAS 2021

Ασκήσεις και Λύσεις

όλα τα επίπεδα

Μετάφραση από τα γερμανικά και
επιστημονική επιμέλεια

Κωνσταντίνος Ζέρβας
Σύμβουλος Εκπαίδευσης Πληροφορικής



έκδοση 1.0

Σύρος 2023

*Μια υψηλού επιπέδου εκπαίδευση στην Πληροφορική εφοδιάζει
τους μαθητές με την ικανότητα να χρησιμοποιούν την
υπολογιστική σκέψη και τη δημιουργικότητα για να
κατανοήσουν και να αλλάξουν τον κόσμο»
The National Curriculum for Computing, England, Department of Education, 2014*

Ο διαγωνισμός Bebras® (προφέρεται “Μπέμπρας” και σημαίνει “Κάστορας” στα Λιθουανικά) ξεκίνησε το 2003 ως ιδέα της Καθηγήτριας Valentina Dagiene του University of Vilnius της Λιθουανίας και υλοποιήθηκε για πρώτη φορά το 2006.

Ο διαγωνισμός “Κάστορας” αποτελεί, σήμερα, μια διεθνή πρωτοβουλία της ομώνυμης κοινότητας (Bebras®), με σκοπό την προώθηση της Πληροφορικής (ή Επιστήμης των Υπολογιστών) και της Υπολογιστικής Σκέψης μεταξύ εκπαιδευτικών, μαθητών και μαθητριών όλων των ηλικιών.

Στον διαγωνισμό συμμετέχουν κάθε χρόνο χιλιάδες μαθητές και μαθήτριες, από πολλές χώρες (www.bebbras.org, www.bebbras.gr). Στην Ελλάδα ο διαγωνισμός διοργανώθηκε με μεγάλη επιτυχία για δύο σχολικά έτη 2018-2019 και 2019-2020, από το Εργαστήριο Μαθησιακής Τεχνολογίας και Διδακτικής Μηχανικής (LTEE lab) του Πανεπιστημίου Αιγαίου, υπό την εποπτεία του Καθηγητή Γιώργου Φεσάκη με τη συνεργασία σημαντικών επιστημονικών-εκπαιδευτικών ενώσεων και πλήθους επιλεγμένων εθελοντών. Στην ιστοσελίδα www.bebbras.org υπάρχουν οι δραστηριότητες και οι λύσεις στην ελληνική γλώσσα των δύο διαγωνισμών.

Τα επόμενα χρόνια ο διαγωνισμός δεν έγινε δυνατόν να πραγματοποιηθεί στην Ελλάδα. Αποφασίσαμε να μεταφράσουμε τις ασκήσεις και τις λύσεις του διαγωνισμού της χρονιάς 2021. Η μετάφραση έγινε από την ελβετική έκδοση και από τη γερμανική γλώσσα. Από την εμπειρία μας διαπιστώσαμε ότι οι δραστηριότητες του διαγωνισμού είναι ενδιαφέρουσες και διασκεδαστικές, είναι κλιμακούμενης δυσκολίας, αρέσουν στους μαθητές και τις μαθήτριες, ενώ ταυτόχρονα αποτελούν ένα καινοτόμο εξαιρετικής ποιότητας εκπαιδευτικό υλικό, ωθώντας τους μαθητές και τις μαθήτριες να ασχοληθούν με την Πληροφορική και να σκεφτούν βαθύτερα σχετικά με την Τεχνολογία. Προσπαθούν επίσης να καλύψουν όσο το δυνατόν περισσότερα θέματα σχετικά με την Υπολογιστική Σκέψη και τον ψηφιακό γραμματισμό. Επιπλέον βοηθούν τους μαθητές να διαβάζουν εκφωνήσεις, να επικεντρώνουν τη σκέψη τους και την προσοχή τους, να κατανοούν και να οργανώνουν τα δεδομένα του προβλήματος, να προσδιορίζουν το τελικό ζητούμενο και να ασκούν τη δημιουργικότητά τους προκειμένου να καταλήξουν στην καλύτερη λύση ή τις λύσεις.

Η Jannet Wing διατύπωσε την άποψη ότι η Υπολογιστική Σκέψη είναι η θεμελιώδης ικανότητα που θα έπρεπε να αναπτύσσουν όλοι οι εγγράμματοι πολίτες στην υποχρεωτική εκπαίδευση συμπληρωματικά με την ανάγνωση, τη γραφή και τα μαθηματικά.

Με πολλή μεγάλη χαρά προσφέρουμε στους/στις εκπαιδευτικούς μας και στους/στις μαθητές/τριές μας τη συλλογή αυτή των ασκήσεων και τους ευχόμαστε καλή και χαρούμενη μελέτη!

Για οποιαδήποτε παρατήρηση ή πρόταση βελτίωσης παρακαλούμε επικοινωνήστε μαζί μας (konzervas@sch.gr).

Κωνσταντίνος Ζέρβας
Σύμβουλος Εκπαίδευσης Πληροφορικής
Κυκλάδες-Πειραιάς

Διαγωνισμός BEBRAS 2021

Ασκήσεις και Λύσεις

Όλα τα επίπεδα

Ελβετική Ένωση Πληροφορικής στην Εκπαίδευση SVIA
(Schweizerischer Verein für Informatik in der Ausbildung)
<https://www.informatik-biber.ch/>

Εκδότες:

**Susanne Datzko, Fabian Frei,
Jean-Philippe Pellet**

Μετάφραση από τα γερμανικά και
επιστημονική επιμέλεια

Κωνσταντίνος Ζέρβας
Σύμβουλος Εκπαίδευσης Πληροφορικής

Σύρος 2023

Συμμετοχές στο Informatik-Biber 2021

Masiar Babazadeh, Susanne Datzko, Fabian Frei, Martin Guggisberg, Gabriel Parriaux, Jean-Philippe Pellet

Διεύθυνση Projekt: Nora A. Escherle

Θερμές ευχαριστίες για την δημιουργία των ασκήσεων για τον ελβετικό διαγωνισμό στους: Juraj Hromkovič, Michael Barot, Christian Datzko, Jens Gallenbacher, Dennis Komm, Regula Lacher, Peter Rossmanith: ETH Zürich, Ausbildunges- und Beratungszentrum für Informatikunterricht Bernadette Spieler: Pädagogische Hochschule Zürich

Η επιλογή των ασκήσεων έγινε σε συνεργασία με τους οργανωτές του Bebras σε Deutschland, Österreich, Ungarn, Slowakei und Litauen. Ιδιαίτερα ευχαριστούμε: Valentina Dagiene, Tomas Šiaulys, Vaidotas Kinčius: Bebras.org

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

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

Gerald Futschek, Florentina Voboril: Technische Universität Wien

Zsuzsa Pluhár: ELTE Informatikai Kar, Ungarn

Michal Winzcer: Comenius University, Slowakei

Η Online-έκδοση του διαγωνισμού υλοποιήθηκε στο cuttle.org. Για την καλή συνεργασία ευχαριστούμε:

Eljakim Schrijvers, Justina Dauksaite, Arne Heijenga, Dave Oostendorp, Andrea Schrijvers, Alieke Stijf, Kyra Willekes: cuttle.org, Niederlande

Chris Roffey: UK Bebras Administrator, Vereinigtes Königreich

Για την υποστήριξη στη διάρκεια του διαγωνισμού ευχαριστούμε: Hanspeter Erni: Schulleitung Sekundarschule Rickenbach

Christoph Frei: Chragokyberneticks (Logo Informatik-Biber Schweiz)

Dr. Andrea Leu, Maggie Winter, Brigitte Manz-Brunner: Senarclens Leu + Partner AG

Αυτό το τεύχος είναι αφιερωμένο στη μνήμη του Martin Guggisberg.

Η γερμανική έκδοση των ασκήσεων χρησιμοποιήθηκε με παρόμοιο τρόπο στη Γερμανία και την Αυστρία.

Η μετάφραση στα γαλλικά έγινε από την Elsa Pellet και η μετάφραση στα ιταλικά από τον Christian Giang.





INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Ο διαγωνισμός Bebras 2021 διοργανώθηκε από την Ελβετική Ένωση Πληροφορικής στην Εκπαίδευση SVIA (Schweizerischer Verein für Informatik in der Ausbildung) και υποστηρίζεται από το Ίδρυμα Hasler

HASLERSTIFTUNG

Η δημιουργία του τεύχους στη γερμανική γλώσσα ολοκληρώθηκε 24 Αυγούστου 2022 με το σύστημα στοιχειοθεσίας LATEX. Θα θέλαμε να ευχαριστήσουμε τον Christian Datzko για την ανάπτυξη και την πολυετή συντήρηση του συστήματος για τη δημιουργία των 36 εκδόσεων αυτού του φυλλαδίου (ανά γλώσσα και επίπεδο σχολείου). Το σύστημα επαναπρογραμματίστηκε με τον ίδιο τρόπο όπως το προηγούμενο σύστημα, το οποίο αναπτύχθηκε μαζί με τον Ivo Blöchliger από το 2014. Ευχαριστούμε τον Jean-Philippe Pellet για την ανάπτυξη της αλυσίδας εργαλείων bebras, η οποία χρησιμοποιείται για την αυτοματοποιημένη μετατροπή εγγράφων Markdown και YAML από το 2020.

Παρατήρηση: Όλοι οι ηλεκτρονικοί σύνδεσμοι επαληθεύτηκαν την 1. Δεκεμβρίου 2021



Οι ασκήσεις αδειοδοτούνται βάσει της Αναφοράς Creative Commons–
**Αναφορά Δημιουργού – Μη Εμπορική Χρήση – Παρόμοια
Διανομή 4.0 (CC-BY-NC-SA)**

Οι συγγραφείς αναφέρονται στη σελίδα 114.



Πρόλογος στην ελβετική έκδοση

Ο διαγωνισμός «Informatik-Biber», που έχει καθιερωθεί εδώ και αρκετά χρόνια σε διάφορες χώρες του κόσμου, έχει στόχο να κεντρίσει το ενδιαφέρον παιδιών και νέων για την πληροφορική. Ο διαγωνισμός διεξάγεται στην Ελβετία στα γερμανικά, γαλλικά και ιταλικά από την Ελβετική Ένωση για την Επιστήμη των Υπολογιστών στην Εκπαίδευση (SVIA) και υποστηρίζεται από το Ίδρυμα Hasler ως μέρος του προγράμματος χρηματοδότησης FIT in IT.

Το Informatik-Biber είναι ο Ελβετός εταίρος της πρωτοβουλίας διαγωνισμού «Bebras International Contest on Informatics and Computer Fluency» (<https://www.bebas.org/>), που ξεκίνησε στη Λιθουανία.

Ο διαγωνισμός πραγματοποιήθηκε στην Ελβετία για πρώτη φορά το 2010. Το 2012 πραγματοποιήθηκε για πρώτη φορά ο «Μικρός Κάστορας» (Επίπεδα 3 και 4).

Ο διαγωνισμός Bebras ενθαρρύνει τους/τις μαθητές/τριες να ασχοληθούν ενεργά με θέματα επιστήμης υπολογιστών. Θέλει να καταρρίψει τους φόβους για την επιστήμη των υπολογιστών ως σχολικό μάθημα και να προκαλέσει ενδιαφέρον για ερωτήσεις σχετικά με αυτή την επιστήμη. Ο διαγωνισμός δεν απαιτεί ειδικές γνώσεις για τον τρόπο χρήσης του υπολογιστή - εκτός από το «σερφάρισμα» στο Διαδίκτυο, γιατί ο διαγωνισμός γίνεται διαδικτυακά στον υπολογιστή. Η δομημένη και λογική σκέψη, αλλά και η φαντασία είναι απαραίτητη για τις ασκήσεις. Οι ασκήσεις έχουν σχεδιαστεί σκόπιμα για περαιτέρω εργασία με την επιστήμη των υπολογιστών πέρα από τον ανταγωνισμό.

Ο διαγωνισμός Informatik-Biber 2021 πραγματοποιήθηκε σε πέντε ηλικιακές ομάδες:

- Τάξη 3 και 4 («Little Biber»)
- Τάξη 5 και 6
- Τάξη 7 και 8
- Τάξη 9 και 10
- Τάξη 11 έως 13

Στις ηλικιακές ομάδες 3 και 4 έπρεπε να λυθούν 9 εργασίες, συγκεκριμένα τρεις από τα τρία επίπεδα δυσκολίας εύκολη, μεσαία και δύσκολη. Για τις ηλικιακές ομάδες 5 και 6 υπήρχαν τέσσερις εργασίες από κάθε επίπεδο δυσκολίας, δηλαδή 12 συνολικά. Για τις υπόλοιπες ηλικιακές ομάδες υπήρχαν 15 εργασίες, δηλαδή πέντε εργασίες ανά επίπεδο δυσκολίας.

Δίνονταν βαθμοί για κάθε σωστή απάντηση και αφαιρούνταν βαθμοί για κάθε λανθασμένη απάντηση. Εάν η ερώτηση δεν απαντηθεί, ο λογαριασμός πόντων παρέμενε αμετάβλητος. Ανάλογα με το επίπεδο δυσκολίας, πιστώθηκε ή αφαιρέθηκε διαφορετικός αριθμός πόντων:



	εύκολο	μέτριο	δύσκολο
Σωστή απάντηση	6 πόντοι	9 πόντοι	12 πόντοι
Λάθος απάντηση	-2 πόντοι	-3 πόντοι	-4 πόντοι

Αυτό το διεθνώς χρησιμοποιούμενο σύστημα απονομής βαθμών έχει σκοπό να εξαλείψει το κίνητρο για τυχαία επιλογή των λύσεων.

Στην αρχή, κάθε συμμετέχων/ουσα είχε 45 πόντους ("Little Beaver": 27 πόντοι, τάξεις 5 και 6: 36 πόντοι) στον λογαριασμό του.

Αυτό επιτρέπει την επίτευξη μέχρι 180 πόντους ("Little Beaver": 108 πόντοι, τάξεις 5 και 6: 144 πόντοι) και το ελάχιστο αποτέλεσμα είναι 0 πόντοι.

Για πολλές εργασίες, οι εναλλακτικές απαντήσεις στην οθόνη εμφανίζονταν με τυχαία σειρά. Οι Ασκήσεις τέθηκαν σε διαφορετικές ηλικιακές ομάδες.

Επιπλέον Πληροφορίες

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/>



Περιεχόμενα

1. Η σφραγίδα του Μιχάλη	9
2. Το σωστό μπλουζάκι.....	13
3. Φτιάχνοντας γέφυρες.....	15
4. Το αγαπημένο δώρο	17
5. Μπρελόκ.....	19
6. Το δένδρο πέφτει !.....	23
7. Το μονοπάτι της χελώνας.....	25
8. Νερό στο μύλο	29
9. Παιχνίδι με μπίλιες	33
10. Σάκος με νομίσματα.....	35
11. Θα συναντηθούν?.....	39
12. Dottis	43
13. Κλέβοντας τη φράουλα.....	47
14. Παρατηρώντας τα ζώα	50
15. Αίνιγμα στα γενέθλια	53
16. Το αγαπημένο δώρο.....	57
17. Σώσε το δέντρο!.....	61
18. Βιβλιοθήκη.....	65
19. Σχέδια από πλακίδια	67
20. SOS από τα βουνά	69
21. Στρώματα κατά πυκνότητα!	73
22. Είναι επείγον!	77
23. Τα δίχτυα της Θέκλας	81



24. Φρούτο με φρούτο.....	85
25. Η μαιμουδίτσα Κοκο σκαρφαλώνει	89
26. Τρελλο-θρανία	93
27. Η ταινία με τις μπίλιες	97
28. Καστορο-projects	101
29. Αριθμο-μπίλιες	103
30. Teamwork	107
31. Μέτρημα με νεύματα	111
32. Ταξινόμηση του κάστορα.....	115
33. Οι οικογένειες της Καστορίας	119





1. Η σφραγίδα του Μιχάλη

Ο Μιχάλης έχει τέσσερις σφραγίδες με διαφορετικά σχήματα. Παίρνει μία σφραγίδα κάθε φορά και με αυτή κάνει δύο σφραγίσματα. Δημιουργείται η ακόλουθη εικόνα:



Ποια σφραγίδα χρησιμοποίησε ο Μιχάλης αρχικά;

A)



B)



C)



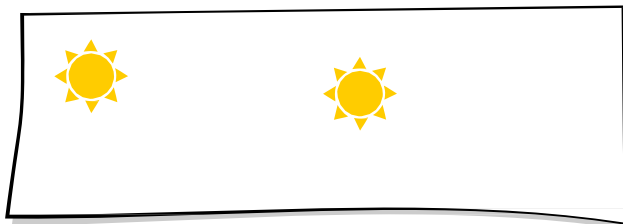
D)



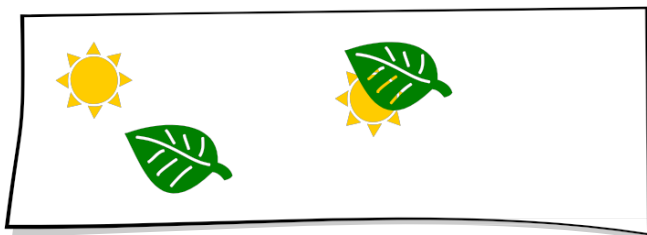
Λύση

Η σωστή απάντηση είναι η C: 

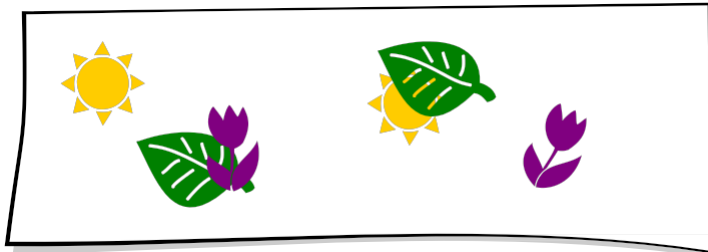
Μπορείτε να διακρίνετε τη σειρά με την οποία ο Μιχάλης χρησιμοποιεί τις σφραγίδες, με το ποιες σφραγίδες είναι πάνω από τις άλλες και ποιες σφραγίδες σκεπάζονται από άλλες. Ο ήλιος σκεπάζεται από όλες τις άλλες σφραγίδες. Άρα η σφραγίδα με τον ήλιο χρησιμοποιήθηκε πρώτη.



Το πράσινο φύλλο σκεπάζει τον ήλιο, αλλά σκεπάζεται τόσο από το λουλούδι όσο και από το σπίτι. Έτσι ο Μιχάλης έχει χρησιμοποιήσει ως δεύτερη σφραγίδα το πράσινο φύλλο.



Το λουλούδι με τη σειρά του επικαλύπτει τόσο το πράσινο φύλλο, όσο και τον ήλιο.



Είναι όμως αδύνατο ο Μιχάλης να χρησιμοποιήσει τελευταία τη σφραγίδα με το λουλούδι. Σε δύο περιπτώσεις το λουλούδι σκεπάζεται από το σπίτι. Έτσι ο Μιχάλης χρησιμοποίησε τη σφραγίδα με το λουλούδι τρίτη στη σειρά. Τελευταία χρησιμοποίησε τη σφραγίδα με το σπίτι.





Αυτό είναι Πληροφορική!

Κατά κάποιον τρόπο, η εικόνα του Μιχάλη είναι μια εικόνα ή μια *μοντελοποίηση* της πραγματικότητας χρησιμοποιώντας σφραγίδες με μοτίβα σε τέσσερα διαφορετικά επίπεδα: ένα επίπεδο με ήλιους, ένα επίπεδο με φύλλα, ένα επίπεδο με λουλούδια και ένα επίπεδο με σπίτια. Η επικάλυψη (το σκέπασμα) των μορφών σε διαφορετικά επίπεδα δίνει τη δυνατότητα στον Μιχάλη να δημιουργήσει στην (δισδιάστατη) επιφάνεια του χαρτιού την ψευδαίσθηση του βάθους και την (τρισδιάστατη) αίσθηση του χώρου.

Κατά τη μοντελοποίηση, συνήθως απεικονίζονται μόνο τα στοιχεία που είναι απαραίτητα για την εκπλήρωση μιας συγκεκριμένης εργασίας ή λειτουργίας. Η πραγματικότητα παρουσιάζεται λοιπόν με απλοποιημένο τρόπο. Η μοντελοποίηση είναι μια σημαντική αρχή στην επιστήμη των υπολογιστών.

Έτσι για παράδειγμα, οι επιστήμονες της Πληροφορικής δημιουργούν ορισμένα προγράμματα με σκοπό να εξερευνήσουν μέρη του πραγματικού κόσμου όσο το δυνατόν γρηγορότερα και με ακρίβεια και να αποκτήσουν γνώσεις για αυτά. Για να δημιουργήσουν ένα τέτοιο πρόγραμμα, οι επιστήμονες υπολογιστών πρέπει πρώτα να σκεφτούν ένα νοητικό μοντέλο που να περιέχει τα στοιχεία του πραγματικού κόσμου που κατά τη γνώμη τους είναι απαραίτητα. Αυτό το νοητικό μοντέλο καθιστά δυνατή τη δημιουργία ενός πραγματικού μοντέλου με τη μορφή ενός διαγράμματος ή ενός προγράμματος υπολογιστή. Με τη βοήθεια αυτού του προγράμματος υπολογιστή, δηλαδή μιας μοντελοποίησης μέρους του πραγματικού κόσμου, μπορεί πλέον να αποκτηθεί γνώση για τον πραγματικό κόσμο.

Λέξεις κλειδιά και websites

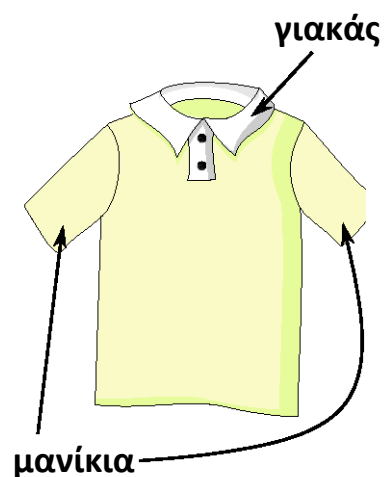
- Μοντελοποίηση δεδομένων (data modeling): https://en.wikipedia.org/wiki/Data_modeling
- Επίπεδα (layers), επεξεργασία εικόνας (digital image editing):
[https://en.wikipedia.org/wiki/Layers_\(digital_image_editing\)](https://en.wikipedia.org/wiki/Layers_(digital_image_editing))
(Ebenen, Bildbearbeitung): <https://de.wikipedia.org/wiki/Ebenentechnik>





2. Το σωστό μπλουζάκι

Η Άννα τακτοποιεί την τσάντα της προκειμένου να πάει στο ματς. Σήμερα πρέπει να φορέσει μπλουζάκι με μανίκια ανοιχτού χρώματος, με σκούρο γιακά αλλά χωρίς να έχει ρίγες.



Ποιο μπλουζάκι θα βάλει στην τσάντα της;



Λύση



Σωστή είναι η απάντηση Β.

Τα μπλουζάκια Α και σήμερα δεν ταιριάζουν επειδή έχουν μαύρα μανίκια, και το μαύρο δεν είναι ανοιχτό χρώμα.

Το μπλουζάκι C έχει ρίγες και γι' αυτό δεν ταιριάζει στο σημερινό ματς.

Το μπλουζάκι Β είναι ιδανικό για σήμερα: Έχει μανίκια ανοιχτού χρώματος, μαύρο γιακά και δεν έχει ρίγες.

Αυτό είναι Πληροφορική!

Σε αυτή την άσκηση πρέπει να διαλέξεις μέσα από ένα σύνολο αντικειμένων, το αντικείμενο που πληροί ή δεν πληροί συγκεκριμένες *συνθήκες (conditions)*.

Εδώ καθορίζονται περισσότερες από μία υπο-συνθήκες (επιμέρους συνθήκες) όπως για παράδειγμα το χρώμα από τα μανίκια και το σχέδιο του υφάσματος και συνδέονται σε μία *συνολική συνθήκη*. Οι επιστήμονες της Πληροφορικής χρησιμοποιούν για αυτές τις συνδέσεις *λογικούς τελεστές (logical operators)*.

Όταν πρέπει να ικανοποιούνται όλες οι συνθήκες ταυτόχρονα τότε χρησιμοποιείται ο τελεστής *KAI (AND)*: Το χρώμα των μανικιών να είναι ανοιχτό *KAI* το χρώμα του γιακά να είναι μαύρο. Εάν αρκεί να ικανοποιείται μία μόνο συνθήκη χρησιμοποιείται ο τελεστής *Η (OR)*. Όταν μια συνθήκη δεν πρέπει να ικανοποιείται, χρησιμοποιούμε τον τελεστή *ΟΧΙ (NOT)*, για παράδειγμα: *ΟΧΙ* (το μπλουζάκι έχει ρίγες), δηλαδή το μπλουζάκι να μην έχει ρίγες.

Για το ψάξιμο (αναζήτηση) σε βάσεις δεδομένων μπορεί κανείς να διατυπώσει πολύπλοκες συνθήκες με τη βοήθεια ειδικών γλωσσών για τη διατύπωση ερωτημάτων (π.χ. δομημένη γλώσσα ερωτοαποκρίσεων-Structured Query Language, SQL).

Οι συνθήκες πρέπει να είναι σαφώς καθορισμένες. Η συνθήκη για παράδειγμα ότι τα μανίκια πρέπει να είναι ανοιχτού χρώματος μπορεί να είναι ασαφής. Σε αυτή την περίπτωση οι προγραμματιστές φτιάχνουν ένα πρόγραμμα ή ένα υποπρόγραμμα (ή συνάρτηση) το οποίο εξετάζει εάν ένα χρώμα είναι ανοιχτό ή όχι ανοιχτό. Εδώ οι προγραμματιστές χρειάζονται έναν σαφή ορισμό πότε ένα χρώμα είναι «ανοιχτό», αλλιώς δεν είναι δυνατόν να γραφεί ένα πρόγραμμα που να λειτουργεί σωστά.

Λέξεις κλειδιά και websites

- Άλγεβρα Boole (Boolean Algebra): https://en.wikipedia.org/wiki/Boolean_algebra
https://de.wikipedia.org/wiki/Boolesche_Algebra
- Λογικές πράξεις (logic operations, logische Verknüpfungen):
https://de.wikipedia.org/wiki/Logische_Verknuepfung



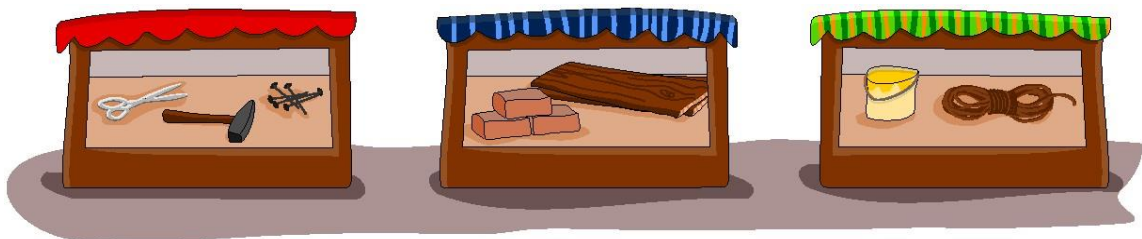
3. Φτιάχνοντας γέφυρες

Η Bella θέλει να φτιάξει μια γέφυρα πάνω από ένα ρυάκι. Χρειάζεται: σφυρί, καρφιά, σανίδες και σκοινί. Στο υπόγειο βρήκε ένα σφυρί και σκοινί.



Τα άλλα πράγματα πρέπει να τα αγοράσει. Παρακάτω βλέπεις τρία καταστήματα καθώς και τα πράγματα που πουλάνε.

Πού μπορεί να αγοράσει η Bella τα πράγματα που δεν έχει;



Λύση

Αυτό είναι το σωστό:



Αυτό είναι Πληροφορική!

Τα καταστήματα σε αυτή την άσκηση πουλάνε συνολικά επτά πράγματα: ψαλίδι, σφυρί, καρφιά, τούβλα, σανίδες, σκοινί και κουβά. Πολλά πράγματα! Τα δύο πράγματα που η Bella χρειάζεται να αγοράσει είναι υποσύνολο αυτού του συνόλου. Αυτό μπορεί κανείς να το ζωγραφίσει ως εξής: Ζωγραφίζουμε το Σύνολο των πραγμάτων και σημειώνουμε με ✓ εάν το κάθε πράγμα ανήκει ή όχι στο υποσύνολο των πραγμάτων που πρέπει να αγοράσει η Bella:



Με τον ίδιο τρόπο μπορεί κανείς να ζωγραφίσει τι πουλάνε τα καταστήματα, για παράδειγμα το κατάστημα αριστερά:



Έτσι βλέπει κανείς εύκολα τι μπορεί να αγοράσει η Bella από το αριστερό κατάστημα: τα καρφιά έχουν ένα πράσινο τικ τόσο στο υποσύνολο των πραγμάτων-προς-αγορά όσο και στο υποσύνολο των πραγμάτων-προς- πώληση.

Επίσης τα προγράμματα στους υπολογιστές πρέπει συχνά να κάνουν συγκρίσεις μεταξύ συνόλων. Αυτό μπορούν να το κάνουν όπως δείξαμε παραπάνω: Για κάθε αντικείμενο χρειάζεται ένα bit. Σε ένα bit ο υπολογιστής μπορεί να αποθηκεύσει μία από τις δύο τιμές, όπως «ΝΑΙ» και «ΟΧΙ». Σε αυτήν την περίπτωση, αποθηκεύεται εάν το στοιχείο ανήκει σε ένα σύνολο («ΝΑΙ») ή όχι («ΟΧΙ»). Στη συνέχεια, ένα πρόγραμμα μπορεί να συγκρίνει δύο σύνολα ως εξής: Ελέγχει εάν το bit για ένα πράγμα σε ένα σύνολο είναι «ΝΑΙ» και το bit για το ίδιο πράγμα στο άλλο σύνολο είναι επίσης «ΝΑΙ». Ένας υπολογιστής μπορεί να εκτελέσει έναν τέτοιο έλεγχο σύγκρισης δύο bit ιδιαίτερα γρήγορα. Στην επιστήμη των υπολογιστών, η περιγραφή συνόλων με bit είναι για αυτό το λόγο πολύ διαδεδομένη.

Λέξεις κλειδιά und websites

- Σύνολο (set): https://en.wikipedia.org/wiki/Set_theory
(Menge): [https://de.wikipedia.org/wiki/Menge_\(Datenstruktur\)](https://de.wikipedia.org/wiki/Menge_(Datenstruktur))
- bits, ακολουθίες bit (bit strings): <https://en.wikipedia.org/wiki/Bit>,
<https://de.wikipedia.org/wiki/Bit>, <https://de.wikipedia.org/wiki/Bitkette>



4. Το αγαπημένο δώρο

Η καστορο-οικογένεια έχει τρία δώρα για τα τρία παιδιά της. Κάθε παιδί λέει πρώτο το πιο αγαπημένο του δώρο και αμέσως μετά το δεύτερο πιο αγαπημένο. Τα δώρα πρέπει να μοιραστούν με τον σωστό τρόπο:

1. Όσο το δυνατόν πιο πολλά παιδιά πρέπει να πάρουν το πιο αγαπημένο τους δώρο (την 1^η προτίμησή τους).
2. Αν κάποιο παιδί δεν πάρει το πιο αγαπημένο του δώρο (την 1^η προτίμησή του), θα πάρει το δεύτερο πιο αγαπημένο του δώρο (την δεύτερη προτίμησή του)

Δώσε στα παιδιά το σωστό δώρο.



1: , 2: 



1: , 2: 

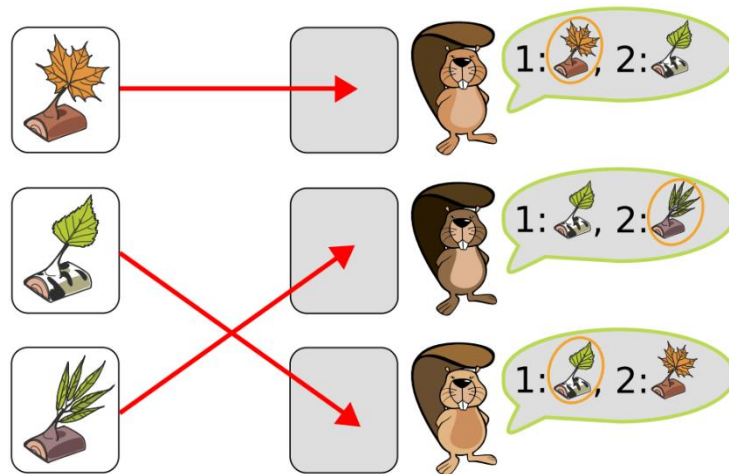


1: , 2: 



Λύση

Εδώ είναι το μόνο ταίριασμα δώρων που πληροί και τις δύο συνθήκες



Μόνο ο δεύτερος κάστορας θέλει το τρίτο δώρο, οπότε πρέπει να το πάρει. Διαφορετικά, θα έπαιρνε κάτι που δεν είναι ούτε το αγαπημένο του δώρο ούτε το δεύτερο αγαπημένο του. Για τα άλλα δύο δώρα, ο διαχωρισμός είναι ξεκάθαρος: ο καθένας μπορεί να πάρει το αγαπημένο του δώρο.

Αυτό είναι Πληροφορική!

Εδώ έχουμε να κάνουμε με ένα πρόβλημα μοναδικής αντιστοίχισης: θέλουμε να αντιστοιχίσουμε τα δώρα έτσι ώστε όλα τα παιδιά να πάρουν ένα δώρο και να μην υπάρχει παιδί χωρίς δώρο. Τα παιδιά δεν έχουν μόνο μία επιθυμία, αλλά υποδεικνύουν μια σειρά προτιμήσεων. Τέτοια προβλήματα αντιστοίχισης προτιμήσεων μπορεί να γίνουν πολύ περίπλοκα. Η επιστήμη των υπολογιστών μας βοηθά να λύσουμε τέτοια προβλήματα όσο το δυνατόν πιο γρήγορα.

Μια δυνατότητα είναι να δώσουμε στις αντιστοιχίσεις μια τιμή: το αγαπημένο δώρο έχει τιμή 1 και το δεύτερο αγαπημένο δώρο έχει τιμή 2. Θέλουμε να ελαχιστοποιήσουμε τη συνολική τιμή. Μια αντιστοίχιση (στα Αγγλικά *matching*) είναι η βέλτιστη εάν δεν υπάρχει άλλη αντιστοίχιση με περισσότερες εκπληρωμένες πρώτες αντιστοιχίσεις. Στην επιστήμη των υπολογιστών, μια τέτοια εργασία αναφέρεται ως *Rank-Maximal-Ranking*. Υπάρχουν πολλά προβλήματα αντιστοίχισης (*Matching-Problems*). Ένα από αυτά ονομάζεται πρόβλημα του σταθερού παντρέματος (*Stable Marriage Problem*). Ακούγεται ενδιαφέρον; Η επιστήμη των υπολογιστών είναι ένα πολύπλευρο επιστημονικό πεδίο.

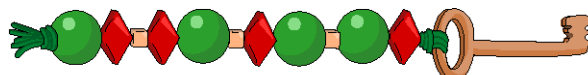
Λέξεις κλειδιά και Websites

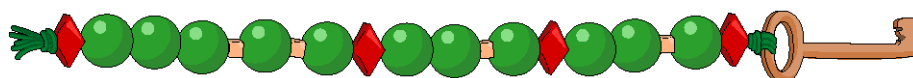
- Zuordnungsproblem: <https://de.wikipedia.org/wiki/Zuordnungsproblem>
- Rang-Wahl: <https://de.wikipedia.org/wiki/Rang-Wahl>

5. Μπρελόκ

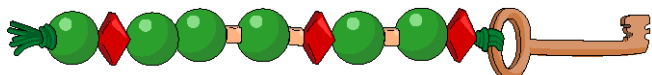
Η ANNA, η BELLA και η LENA φτιάχνουν μπρελόκ με το όνομά τους. Για τα γράμματα χρησιμοποιούν δύο ειδών χάντρες:  και .

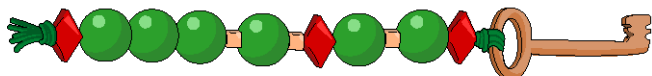
Γράμματα που είναι μόνα τους χωρίζονται με τη χάντρα: 

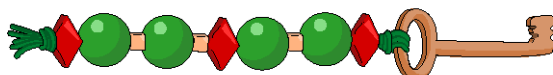
ANNA 

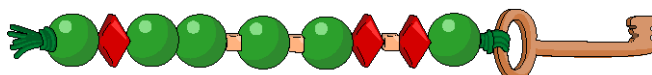
BELLA 

Ποιο μπρελόκ έχει φτιάξει η Λένα;

A) 

B) 

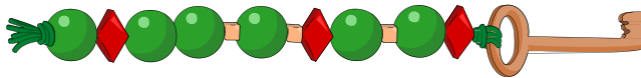
C) 

D) 



Λύση

Η σωστή απάντηση είναι η Α



Η λέξη LENA αρχίζει με L. Στη λέξη BELLA το L είναι το τρίτο και τέταρτο γράμμα, οπότε για το L βλέπουμε στη λέξη BELLA το συνδυασμό χαντρών . Μόνο οι απαντήσεις Α) και D) αρχίζουν με αυτό το γράμμα και θα μπορούσαν να είναι η λύση. Το δεύτερο γράμμα της λέξης LENA, δηλαδή το E, είναι επίσης το δεύτερο γράμμα της λέξης BELLA, οπότε και στα δύο μπρελόκ βρίσκουμε στο μπρελόκ τη χάντρα .

Τόσο στο Α) όσο και στο D) έχουμε το ως δεύτερο γράμμα, οπότε θα μπορούσαν να είναι και τα δύο η λύση. Στη συνέχεια θέλουμε να βρούμε τις χάντρες για το γράμμα N.

Στο μπρελόκ της ANNA βλέπουμε ότι το γράμμα N αντιστοιχεί στο .

Το οποίο υπάρχει ως τρίτο γράμμα στο Α), που είναι και η λύση.

Μια άλλη δυνατότητα για να βρούμε τις χάντρες για το μπρελόκ της LENA θα ήταν να κατασκευάσουμε έναν πίνακα με τους συνδυασμούς χαντρών που έχουν ήδη χρησιμοποιηθεί για τα μπρελόκ της ANNA και BELLA. Από το μπρελόκ της ANNA βρίσκουμε ότι για το Α χρησιμοποιήθηκε ο συνδυασμός , για το N ο συνδυασμός .

Από το μπρελόκ της BELLA βρίσκουμε ότι για το Β χρησιμοποιήθηκε ο συνδυασμός , για το Ε: και για το L: .

Γράμμα	Χάντρες
A	
N	
B	
E	
L	

Έτσι μπορούμε να φτιάξουμε το μπρελόκ της LENA από τις χάντρες , , και .

Για διαχωριστικό χρησιμοποιούμε τη χάντρα .

Έτσι σχηματίζουμε την ακολουθία .

Οπότε η απάντηση Α) είναι η σωστή. Εάν αποκωδικοποιήσουμε τις άλλες πιθανές απαντήσεις με τη βοήθεια του πίνακα κωδικοποίησης βρίσκουμε για το Β) BENA, για το C) NENA και για το D) LEAN.



Αυτό είναι Πληροφορική!

Οι πληροφορίες κωδικοποιούνται για να είναι δυνατή η μετάδοση μηνυμάτων υπό ορισμένες συνθήκες ή για να κρατηθούν οι πληροφορίες μυστικές (κρυπτογράφηση). Σε αυτή την άσκηση, η κωδικοποίηση βασίζεται στον κώδικα Μωρς. Η τελεία • του κώδικα απεικονίζεται με τη χάντρα  και η – με τη χάντρα .

Το γράμμα Α είναι στον κώδικα Μωρς το • — , στον κώδικα με τις χάντρες το  .

Για να κωδικοποιήσουμε οποιοδήποτε κείμενο, χρειαζόμαστε κωδικούς για όλα τα γράμματα του αλφαβήτου. Ο κώδικας Μωρς ξεκίνησε τον 19^ο αιώνα. Ο Samuel Morse εφηύρε έναν απλό ηλεκτρομαγνητικό τηλεγράφο εγγραφής το 1837. Ο κώδικας που χρησιμοποιήθηκε τότε περιελάμβανε μόνο τα δέκα ψηφία 0 έως 9. Οι αριθμοί που εκπέμπονταν έπρεπε να μεταφραστούν σε γράμματα και λέξεις με τη βοήθεια πίνακα. Ο Alfred Lewis Vail, υπάλληλος του Morse, ανέπτυξε τον πρώτο κώδικα το 1838, ο οποίος περιλάμβανε και γράμματα. Ο κώδικας σχεδιάστηκε για να μεταδίδει κείμενο ακουστικά, οπτικά ή ηλεκτρικά μέσω τηλεγραφικών γραμμών. Μια τελεία είναι ένας μικρός χρόνος μετάδοσης και μια παύλα είναι τρεις φορές μεγαλύτερη. Πρέπει να υπάρχει μια παύση μεταξύ των γραμμάτων. Μια μεγαλύτερη παύση χωρίζει τις λέξεις.

Ακόμη και σήμερα χρησιμοποιείται ο κώδικας Μωρς για το σήμα SOS. Το σήμα SOS σε κώδικα Μωρς ••• — — — ••• (3 σύντομα, 3 αργά, 3 σύντομα) μπορεί να υλοποιηθεί πολύ εύκολα με φωνή, με χτυπήματα ή με φακό. Στην ηλεκτρονική επεξεργασία δεδομένων, οι χαρακτήρες κωδικοποιούνται χρησιμοποιώντας μια αριθμητική τιμή για τη μετάδοση ή την αποθήκευσή τους

Λέξεις κλειδιά και Websites

- κωδικοποίηση χαρακτήρων (character encoding):
https://en.wikipedia.org/wiki/Character_encoding
Zeichenkodierung: <https://de.wikipedia.org/wiki/Zeichenkodierung>
- Morse code: <https://de.wikipedia.org/wiki/Morsecode>





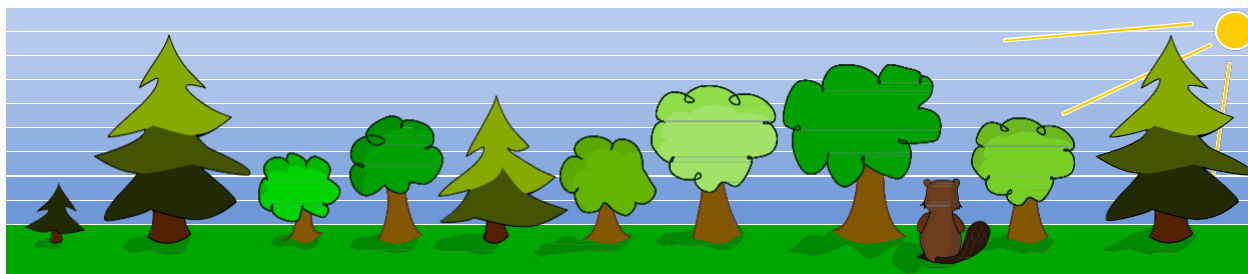
6. Το δέντρο πέφτει !

Ένας κάστορας θέλει να φτιάξει ένα φράγμα. Για να πέφτουν πάντα τα σωστά δέντρα, σκέφθηκε δύο συνθήκες:

Ένα δέντρο πέφτει, μόνο όταν ...

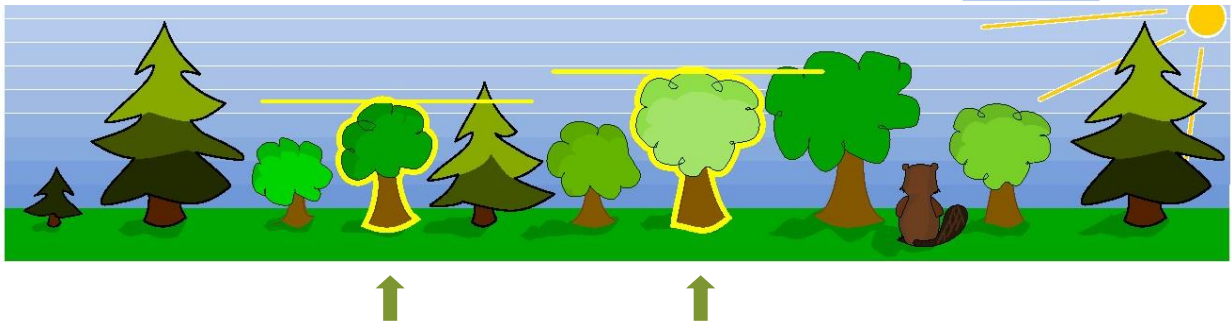
- ... ακριβώς αριστερά μεγαλώνει ένα μικρότερο δέντρο και ...
- ... ακριβώς δεξιά μεγαλώνει ένα μεγαλύτερο δέντρο.

Ποια δέντρα θα κόψει ο κάστορας?



Λύση

Μόνο τα δέντρα στην τέταρτη και έβδομη θέση πληρούν τις δεδομένες συνθήκες: υπάρχει ένα μικρό δέντρο ακριβώς στα αριστερά ΚΑΙ ένα μεγαλύτερο δέντρο ακριβώς στα δεξιά.



Αυτό είναι Πληροφορική!

Η επιστήμη των υπολογιστών συχνά ασχολείται με την επίλυση προβλημάτων που καθορίζονται από μια σειρά λογικών περιορισμών (*constraints*). Το ζητούμενο είναι να βρεθεί μια λύση που να ικανοποιεί όλους τους δεδομένους περιορισμούς. Πιο πολύπλοκα προβλήματα μπορούν να λυθούν όταν οι περιορισμοί περιγραφούν συνδυάζοντας τους λογικούς τελεστές. Για παράδειγμα, η πράξη του λογικού πολλαπλασιασμού ($\wedge$ ή τελεστής AND) στην έκφραση $A \wedge B$ επιστρέφει «true» («αληθής») ως αποτέλεσμα εάν και μόνο εάν και οι δύο περιορισμοί είναι «true» (αληθείς). Έτσι, σε αυτήν την εργασία θα είχαμε: «Το δέντρο στα αριστερά είναι μικρότερο» $\wedge$ «το δέντρο στα δεξιά είναι μεγαλύτερο». Αυτή η βασική αρχή ισχύει σχεδόν σε όλους τους τομείς της επιστήμης των υπολογιστών, όπως σε πολλούς αλγόριθμους ταξινόμησης, για παράδειγμα στον αλγόριθμο *Bubble Sort* (ταξινόμηση φυσαλίδας), όπου πολλά αντικείμενα σε έναν πίνακα ελέγχονται πάντα για ορισμένους περιορισμούς, ώστε στη συνέχεια, εάν είναι απαραίτητο, να μετακινηθούν σε διαφορετική θέση στον πίνακα. Αυτή η αρχή επαναλαμβάνεται μέχρι να ταξινομηθεί πλήρως ο πίνακας.

Λέξεις κλειδιά και Websites

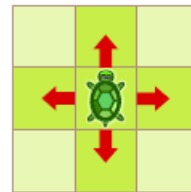
- Αλγοριθμική σκέψη, *algorithmic thinking*, *Algorithmisches Denken*
- λογικοί τελεστές, *logical operators*, *logische Operatoren*:
https://de.wikipedia.org/wiki/Logischer_Operator
- Ταξινόμηση, *sort*, *Sortieren*: <https://sorting.at/>
- Πρόβλημα ικανοποίησης περιορισμών, *Constraint Satisfaction Problem*:
<https://de.wikipedia.org/wiki/Constraint-Satisfaction-Problem>



7. Το μονοπάτι της χελώνας

Μια χελώνα μπορεί να βοσκήσει σε διαφορετικούς κήπους. Κάθε κήπος χωρίζεται σε τετράγωνα καλυμμένα είτε με γρασίδι είτε με πέτρες.

Η χελώνα δεν μπορεί να περπατήσει στις πέτρες. Αλλά μπορεί να μετακινηθεί από το ένα τετράγωνο με γρασίδι στο άλλο, που βρίσκεται ακριβώς δίπλα της.



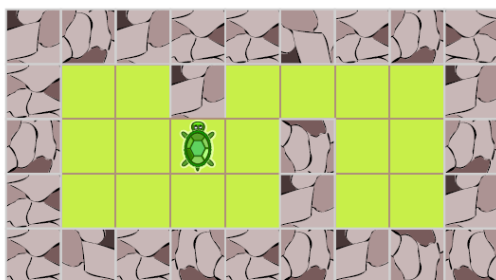
Η χελώνα πρέπει να βοσκήσει όλον τον κήπο. Ξεκινά από το τετράγωνο όπου φαίνεται στην εικόνα.

Στο τέλος, πρέπει να έχει περάσει από κάθε τετράγωνο μόνο μία φορά.

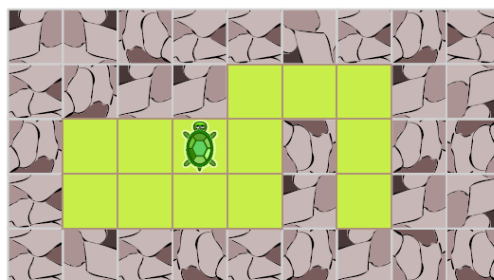
Δυστυχώς η χελώνα, έναν κήπο δεν μπόρεσε να τον βοσκήσει σε όλη του την έκταση.

Ποιος ήταν αυτός;

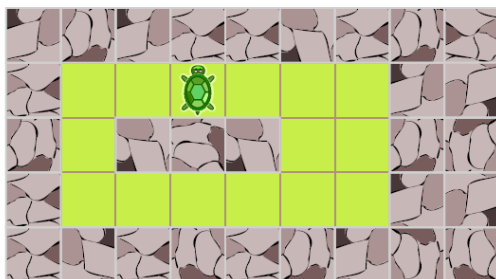
A)



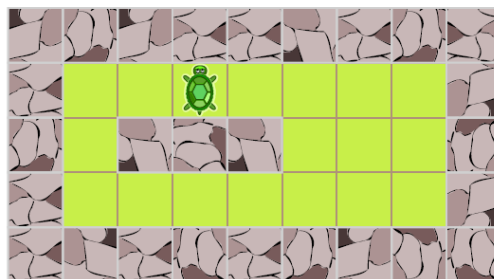
B)



C)



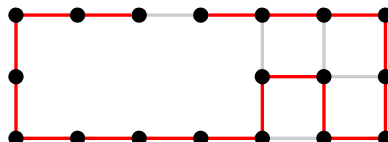
D)



Αυτό είναι Πληροφορική!

Η χελώνα πρέπει να ακολουθήσει ένα δρόμο μέσα στον κήπο της έτσι ώστε να επισκεφθεί κάθε τετράγωνο ακριβώς μία φορά. Το πρόβλημα πίσω από αυτή την άσκηση είναι το λεγόμενο *Hamiltonian path problem*.

Ένας κήπος δηλαδή τα τετράγωνα με γρασίδι μπορούν να αναπαρασταθούν ως εξής: Κάθε τετράγωνο με γρασίδι είναι ένας κόμβος (απεικονίζεται με τελεία). Ο κήπος φαίνεται ως εξής:



Για τέτοιες δομές (που, από επιστήμονες υπολογιστών αλλά και από μαθηματικούς ονομάζονται γράφοι, Αγγλικά *graphs*), ο Sir William Rowan Hamilton έθεσε το ερώτημα τον 19^ο αιώνα, εάν υπάρχει μονοπάτι σε έναν γράφο που να περνά από κάθε κόμβο αλλά ακριβώς μία φορά. Μια τέτοια διαδρομή ονομάζεται επίσης *μονοπάτι Hamilton* (*Hamiltonian path*). Γενικά, το πρόβλημα της εύρεσης μονοπατιού Hamilton είναι πολύ δύσκολο να λυθεί. Κανείς δεν γνωρίζει έναν *αλγόριθμο* που θα μπορούσε αποτελεσματικά (σε λιγότερο ή περισσότερο λογικό χρόνο) να αποφασίσει για οποιοδήποτε γράφο εάν υπάρχει μονοπάτι Hamilton σε αυτόν το γράφο. Επίσης, δεν γνωρίζουμε αν μπορεί καν να υπάρξει ένας τόσο αποτελεσματικός αλγόριθμος. Αυτό ισχύει για όλα τα λεγόμενα *NP-complete προβλήματα* (*NP-complete problems*), από τα οποία το πρόβλημα του μονοπατιού Hamilton είναι ένα από τα πιο διάσημα.

Λέξεις κλειδιά και Websites

- γράφοι (*graphs*), μονοπάτι Hamilton (*Hamiltonian path*):
https://en.wikipedia.org/wiki/Hamiltonian_path
Graph, Hamiltonpfad: <https://de.wikipedia.org/wiki/Hamiltonkreisproblem>





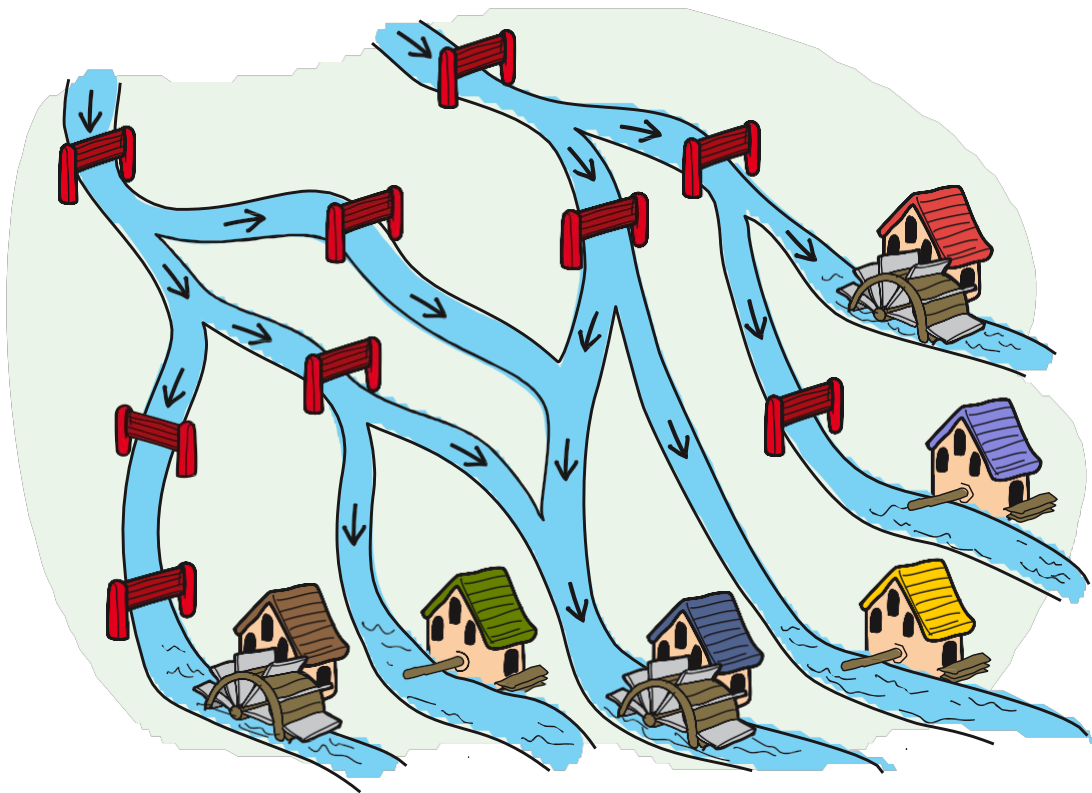


8. Νερό στο μύλο

Ο μυλωνάς κύριος Μήτσος έχει έξι μύλους. Σε τρεις μύλους πρέπει να εγκαταστήσει τον τροχό που λείπει. Για να γίνει αυτό πρέπει στους συγκεκριμένους μύλους να σταματήσει η ροή του νερού. Στους άλλους μύλους όμως το νερό πρέπει να τρέχει.

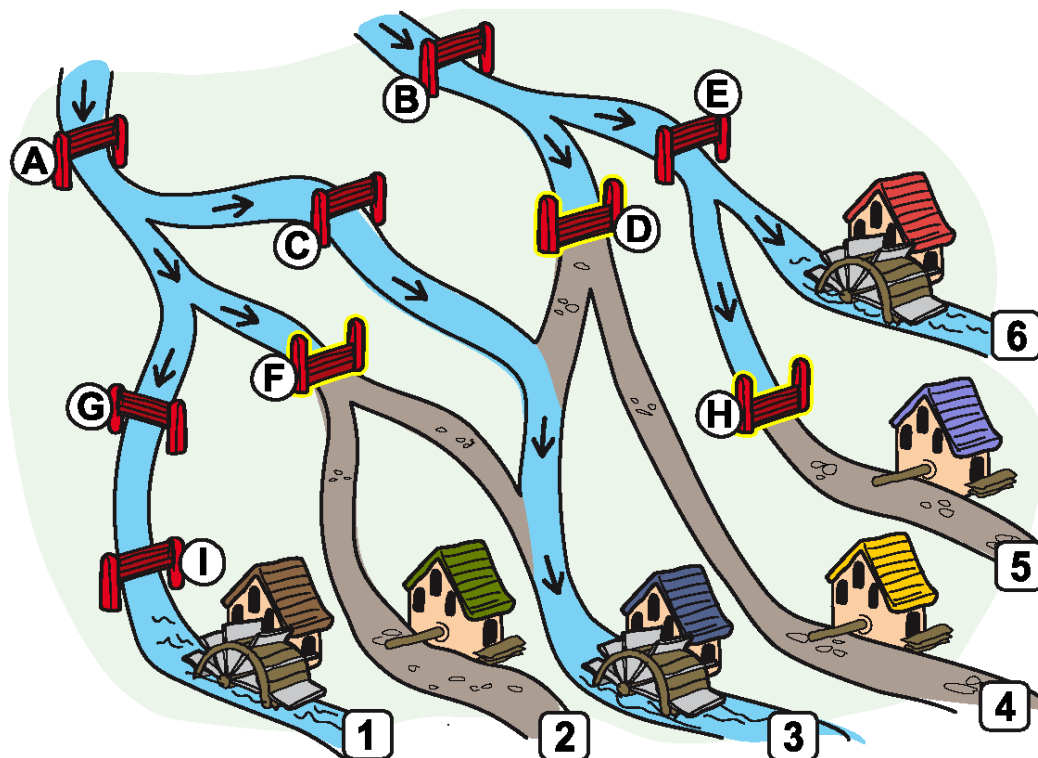
Το νερό τρέχει μόνο προς τα κάτω. Ένας συρόμενος φράχτης σταματά τη ροή του νερού.

Ποιους συρόμενους φράχτες πρέπει να κλείσει ο μυλωνάς Μήτσος;



Λύση

Η σωστή απάντηση είναι: Πρέπει να κλείσουν τρεις συρόμενοι φράχτες, που στο ακόλουθο διάγραμμα έχουν σχεδιαστεί με τα γράμματα D, F και H.



Αυτός είναι ο μόνος τρόπος που το νερό σταματά να ρέει στους μύλους 2, 4 και 5 που δεν έχουν τροχό, ενώ οι μύλοι 1, 3 και 6 συνεχίζουν να δέχονται νερό:

- Οι φράκτες A, G και I πρέπει όλοι να μείνουν ανοικτοί, αλλιώς δεν θα τρέξει νερό στο μύλο 1.
- Οι φράκτες B και E πρέπει επίσης να μείνουν ανοικτοί, αλλιώς δεν θα τρέξει νερό στο μύλο 6.
- Επειδή οι φράκτες B και E παραμένουν ανοικτοί, ο φράκτης H πρέπει επίσης να κλείσει γιατί αλλιώς θα τρέξει νερό στον μύλο 5
- Επειδή ο φράκτης A κλείνει, πρέπει να κλείσει και ο φράκτης F, γιατί αλλιώς θα τρέξει νερό στο μύλο 2
- Επειδή ο φράκτης B παραμένει ανοικτός, πρέπει να κλείσει ο φράκτης D, γιατί αλλιώς θα τρέξει νερό στον μύλο 4
- Επειδή οι φράκτες D και F είναι κλειστοί, πρέπει ο φράκτης C να μείνει ανοικτός, γιατί αλλιώς δεν θα τρέξει νερό στον μύλο 3



Αυτό είναι Πληροφορική!

Σε αυτή την άσκηση, η ροή του νερού στους μύλους ρυθμίζεται από *συνθήκες*. Για παράδειγμα, το νερό ρέει στον μύλο 6 ακριβώς όταν οι δύο συρόμενοι φράκτες B και E είναι ανοικτοί. Και εδώ είναι ένα δεύτερο, κάπως πιο περίπλοκο παράδειγμα: Το νερό ρέει στο μύλο 3 εάν πληρούνται τουλάχιστον μία ή και οι δύο από τις ακόλουθες προϋποθέσεις:

- Ο φράκτης A είναι ανοικτός και ένας από τους δύο φράκτες C και F είναι ανοικτός.
- Οι δύο φράκτες B και D είναι ανοικτοί.

Τέτοιες σύνθετες συνθήκες συνδυάζονται με τους λογικούς τελεστές AND (ΚΑΙ) (ως σύμβολο: $\wedge$) ή OR (Ή) (ως σύμβολο: $\vee$). Τέτοιοι τελεστές συνδυάζουν λογικές τιμές όπως ΑΛΗΘΗΣ (true) ή ΨΕΥΔΗΣ (false). Έτσι, εάν το A και το B είναι δύο λογικές τιμές, μπορείτε να καθορίσετε ποιες λογικές τιμές παίρνουν οι σύνθετες εκφράσεις «A ΚΑΙ B» ή «A Ή B»:

A	B	A ΚΑΙ B	A Ή B
ΨΕΥΔΗΣ	ΨΕΥΔΗΣ	ΨΕΥΔΗΣ	ΨΕΥΔΗΣ
ΑΛΗΘΗΣ	ΨΕΥΔΗΣ	ΨΕΥΔΗΣ	ΑΛΗΘΗΣ
ΨΕΥΔΗΣ	ΑΛΗΘΗΣ	ΨΕΥΔΗΣ	ΑΛΗΘΗΣ
ΑΛΗΘΗΣ	ΑΛΗΘΗΣ	ΑΛΗΘΗΣ	ΑΛΗΘΗΣ

Στην Πληροφορική (αλλά και στα Μαθηματικά) η έκφραση «A ΚΑΙ B» θεωρείται σωστή (ΑΛΗΘΗΣ) όταν και το A και το B είναι ΑΛΗΘΗ. Η έκφραση «Τρέχει νερό στον μύλο 6» έχει το ίδιο νόημα με την έκφραση:

«Ο φράκτης B είναι ανοικτός» ΚΑΙ «Ο φράκτης E είναι ανοικτός».

Η έκφραση «Τρέχει νερό στον μύλο 3», είναι ισοδύναμη με την έκφραση:

(«Ο φράκτης A είναι ανοικτός» ΚΑΙ («Ο φράκτης C είναι ανοικτός» Ή «Ο φράκτης F είναι ανοικτός»)) Ή («Ο φράκτης B είναι ανοικτός» ΚΑΙ «Ο φράκτης D είναι ανοικτός»).

Κάθε AND και κάθε OR συνδέουν δύο δηλώσεις. Οι παρενθέσεις καθορίζουν τη σειρά με την οποία συνδέονται οι εντολές. Κατά τον προγραμματισμό, η σωστή διατύπωση των συνθηκών είναι σημαντική. Ο συνδυασμός λογικών τελεστών και παρενθέσεων είναι χρήσιμος για τη διαμόρφωση πιο περίπλοκων συνθηκών. Και στις εντολές διακλάδωσης με τη χρήση του if, και σε βρόχους με τη χρήση του while, χρησιμοποιούμε συνθήκες για τον έλεγχο της ροής του προγράμματος.

Λέξεις κλειδιά και Websites

- Εντολή υπό συνθήκη, conditional instruction, Bedingte Anweisung:
https://de.wikipedia.org/wiki/Bedingte_Anweisung_und_Verzweigung
- Άλγεβρα Boole (Boolean Algebra): https://en.wikipedia.org/wiki/Boolean_algebra
https://de.wikipedia.org/wiki/Boolesche_Algebra
- Λογικές πράξεις (logic operations, logische Verknüpfungen):
https://de.wikipedia.org/wiki/Logische_Verknuepfung

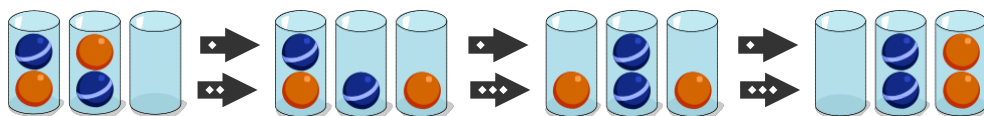




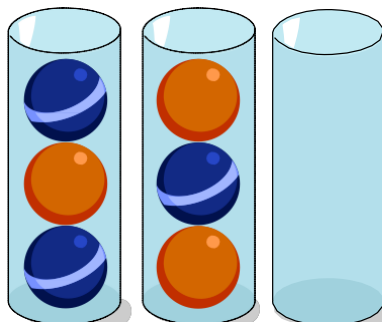
9. Παιχνίδι με μπίλιες

Οι κάστορες θέλουν να ταξινομήσουν μπίλιες ανάλογα με το χρώμα τους. Στο τέλος, όλες οι μπίλιες πρέπει να είναι σε δύο ποτήρια. Σε ένα ποτήρι, πρέπει να είναι οι μπίλιες που έχουν το ίδιο χρώμα. Πρέπει να ακολουθήσετε τους εξής τρεις κανόνες:

- ➡ Κανόνας 1: Σε κάθε βήμα μπορεί να μετακινηθεί μόνο η μπίλια που είναι πάνω πάνω στο ποτήρι
- ➡ Κανόνας 2: Κάθε μπίλια μπορεί να μετακινηθεί σε ένα άδειο ποτήρι
- ➡ Κανόνας 3: Μια μπίλια μπορεί να μετακινηθεί σε ένα ποτήρι, εφόσον υπάρχει θέση στο ποτήρι και η μπίλια που είναι ακριβώς από κάτω έχει το ίδιο χρώμα

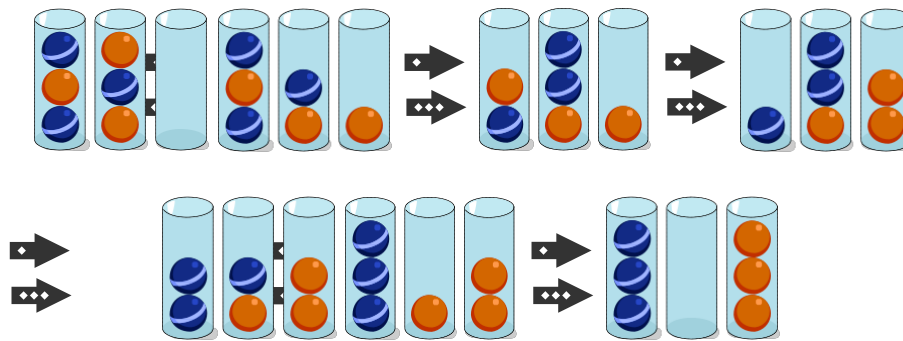


Ταξινόμηση τις μπίλιες τηρώντας τους τρεις κανόνες.



Λύση

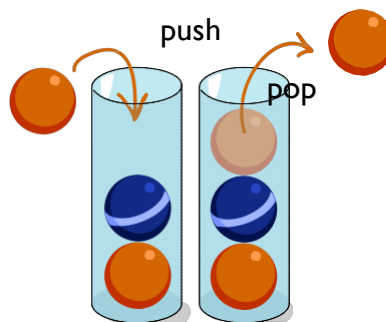
Οι μπίλιες μπορούν να μετακινηθούν με την ακόλουθη σειρά:



Για να αναδιατάξετε τις μπάλες, χρειάζεστε τουλάχιστον 6 βήματα. Υπάρχουν και άλλοι τρόποι για να αναδιατάξετε τις μπάλες σε μόλις 6 βήματα.

Αυτό είναι Πληροφορική!

Σε αυτήν την άσκηση, μετακινείτε τις μπάλες με παρόμοιο τρόπο με τον τρόπο με τον οποίο ο υπολογιστής διαχειρίζεται δεδομένα σε μια *στοίβα* (*stack*): Μπορεί να προσθέσει μόνο ένα στοιχείο (μία μπίλια στην άσκηση) στην κορυφή (ώθηση, αγγλικά *push*) και να αφαιρέσει μόνο το επάνω στοιχείο (αγγλικά *pop*).



Ο υπολογιστής μπορεί να έχει πρόσβαση στα κάτω στοιχεία μόνο εάν αφαιρεθούν πρώτα οι μπίλιες που είναι από πάνω. Και το στοιχείο που αποθηκεύτηκε τελευταίο θα αφαιρεθεί πρώτο. Αυτόν τον κανόνα οι επιστήμονες της Πληροφορικής τον ονομάζουν *Last-in-First-out* κανόνα (εν συντομία *LIFO*).

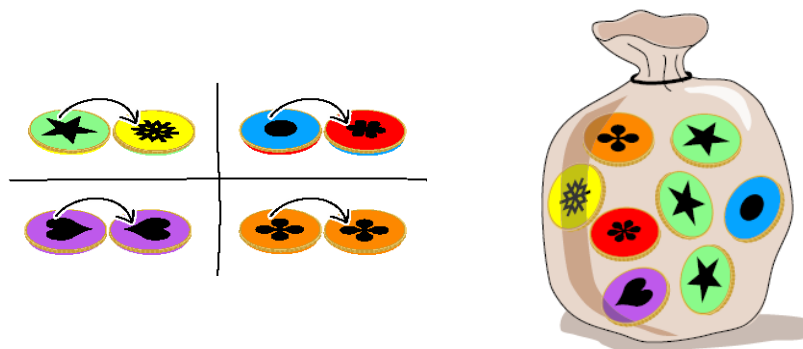
Λέξεις κλειδιά και Websites

- Στοίβα (*stack*): [https://en.wikipedia.org/wiki/Stack_\(abstract_data_type\)](https://en.wikipedia.org/wiki/Stack_(abstract_data_type))
 Stapelspeicher, Kellerspeicher: <https://de.wikipedia.org/wiki/Stapelspeicher>



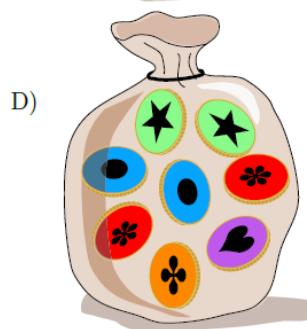
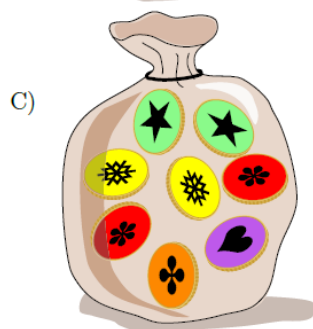
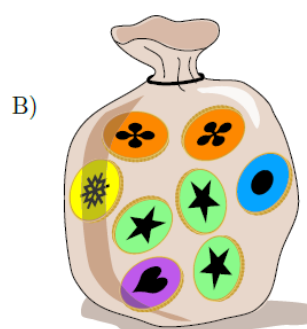
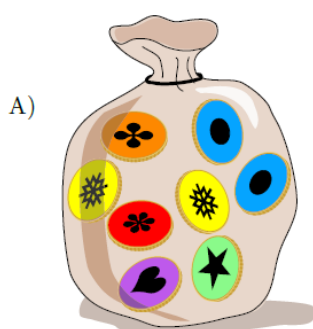
10. Σάκος με νομίσματα

Στη χώρα του Αιμίλιου υπάρχουν 4 διαφορετικά είδη νομισμάτων. Εδώ βλέπεις τις δύο όψεις κάθε νομίσματος καθώς και το σάκο του Αιμίλιου με τα νομίσματά του.



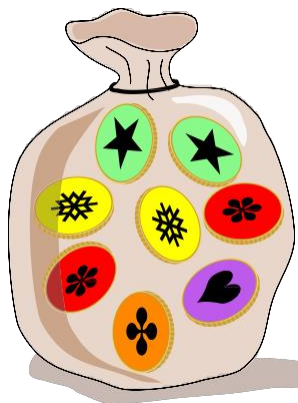
Κάποιος ανακινεί το σάκο, οπότε τα νομίσματα ανακατεύονται.

Ποιος είναι ο σάκος του Αιμίλιου?



Λύση

Η σωστή απάντηση είναι η C:



Στο σάκο του Αιμίλιου βρίσκονται

- 4 νομίσματα τύπου  
- 2 τύπου  
- 1 τύπου  
- 1 τύπου  

	σάκος Αιμίλιου	σάκος Α	σάκος Β	σάκος C	σάκος D
 	4	3	4	4	2
 	2	3	1	2	4
 	1	1	2	1	1
 	1	1	1	1	1

Μόνο ο σάκος C έχει τον ίδιο αριθμό νομισμάτων με τον σάκο του Αιμίλιου για κάθε τύπο νομίσματος. Επομένως, το C είναι η μόνη σωστή λύση.





Αυτό είναι Πληροφορική!

Σε αυτήν την άσκηση πρέπει να αναγνωρίσετε τους τύπους νομισμάτων χωρίς να βλέπετε και τις δύο όψεις των νομισμάτων. Έχετε μόνο ελλιπείς πληροφορίες. Τα αντικείμενα του πραγματικού κόσμου αποθηκεύονται σε ένα σύστημα υπολογιστή με τα βασικά χαρακτηριστικά τους. Συχνά αρκεί να γνωρίζουμε μόνο ένα μέρος αυτών των χαρακτηριστικών για να μπορούμε να αναγνωρίσουμε ένα αντικείμενο.

Μια κάμερα σε ένα όχημα αυτόνομης οδήγησης βλέπει πάντα μόνο μέρη της πραγματικότητας και το σύστημα του υπολογιστή πρέπει να μπορεί να αναγνωρίζει οχήματα, ανθρώπους ή αντικείμενα που βρίσκονται στο δρόμο και να αντιδρά σωστά στην αντίστοιχη κυκλοφοριακή κατάσταση. Η τεχνητή νοημοσύνη στα συστήματα υπολογιστών μαθαίνει σταδιακά και βελτιώνεται στην αναγνώριση ολόκληρων αντικειμένων από θραύσματα, όπως ακριβώς κάνουν οι άνθρωποι.

Λέξεις κλειδιά και Websites

- πολυσύνολο (multiset, bag, mset): <https://en.wikipedia.org/wiki/Multiset>
Multimenge: <https://de.wikipedia.org/wiki/Multimenge>
- μη δομημένα δεδομένα (unstructured data):
https://en.wikipedia.org/wiki/Unstructured_data
unstrukturierte Daten: https://de.wikipedia.org/wiki/Unstrukturierte_Daten

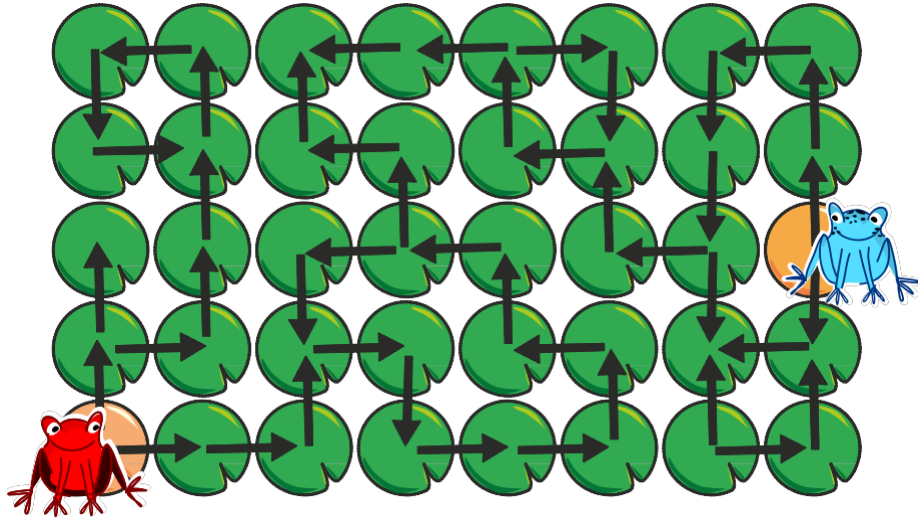




11. Θα συναντηθούν?

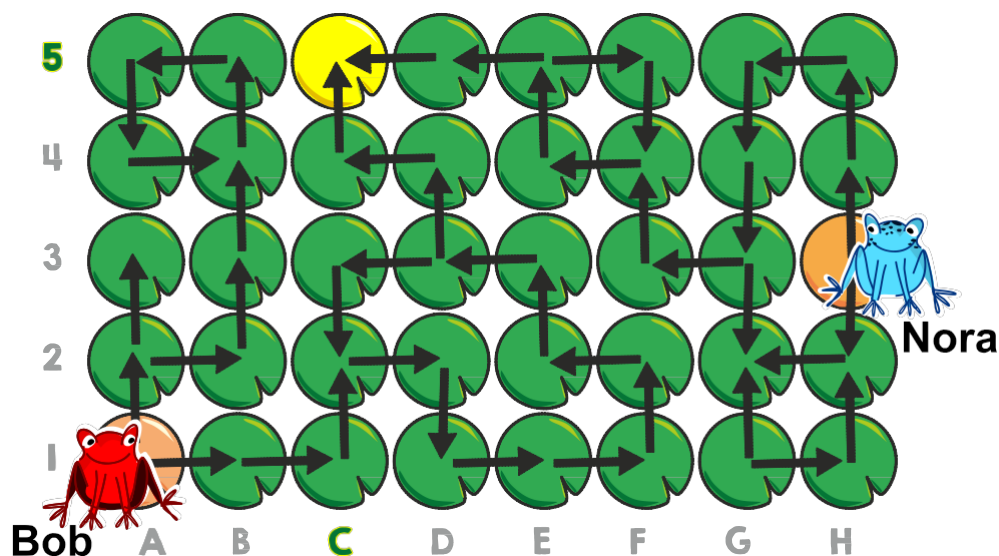
Σε μια λίμνη, δύο βάτραχοι μπορούν να πηδήξουν από νούφαρο σε νούφαρο - αλλά μόνο στην κατεύθυνση που δείχνουν τα βέλη.

Σε ποιο νούφαρο θα συναντηθούν;



Λύση

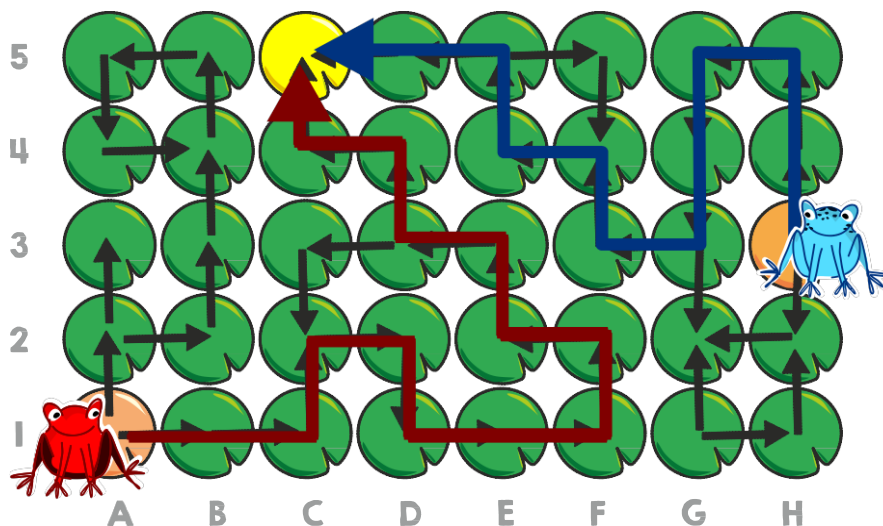
Οι βάτραχοι μπορούν να συναντηθούν μόνο στο νούφαρο C5.



Από την αρχική του θέση, ο κόκκινος βάτραχος Bob έχει δύο επιλογές: Αν πάει «πάνω», είτε θα καταλήξει στο αδιέξοδο A3 είτε θα κολλήσει στο κυκλικό αδιέξοδο που ξεκινά από το B4. Πάει, λοιπόν, στην αρχή "δεξιά" (προς το B1) και μετά μπορεί να πηδήξει έως το D3. Εκεί μπορεί να πηδήξει «αριστερά» σε κυκλικό αδιέξοδο, που τον φέρνει πίσω στο D3, ή «πάνω», που τον πάει πιο μακριά στο C5 – που είναι όμως αδιέξοδο.

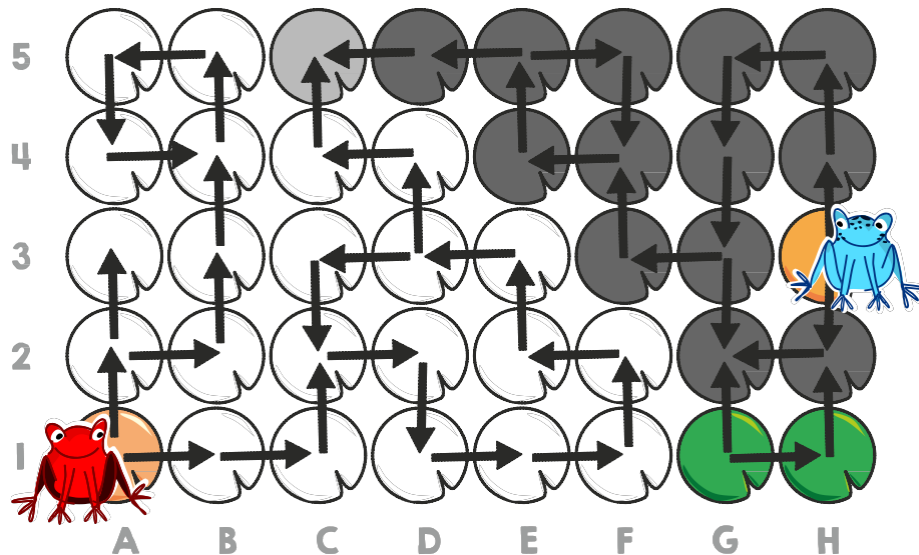
Ο μπλε βάτραχος Nora έχει επίσης δύο επιλογές στην αρχή. Αν πάει «κάτω», μπαίνει στο αδιέξοδο G2. Αν ξεκινήσει «προς τα πάνω», φτάνει πρώτα στο G3. Από εκεί, μπορεί είτε να επιστρέψει στο αδιέξοδο G2 είτε να στρίψει "αριστερά" και τελικά να φτάσει στο E5. Από εκεί εάν πάει «δεξιά» μπαίνει σε κυκλικό αδιέξοδο που οδηγεί πάλι στο E5 ή εάν πάει «αριστερά» φθάνει στο αδιέξοδο στο C5.

Γνωρίζουμε ήδη ότι ο Bob μπορεί επίσης να φτάσει στο C5, άρα μπορούν να συναντηθούν εκεί. Το σχεδιάγραμμα δείχνει πως μπορούν να φτάσουν και οι δύο στο C5.





Αλλά αυτό δεν εγγυάται ότι δεν μπορούν να συναντηθούν και κάπου αλλού. Το επόμενο σχεδιάγραμμα δείχνει όλα τα νούφαρα που μπορούν να φτάσουν ο Μπομπ Bob (λευκό) και η Nora (σκούρο γκρι) αν ακολουθήσουν τα βέλη. Βλέπουμε ότι μπορούν να συναντηθούν μόνο στο C5.



Αυτό είναι Πληροφορική!

Πώς μπορεί να δημιουργηθεί η τελευταία εικόνα; Τα νούφαρα στα οποία μπορεί να φτάσει ένας βάτραχος μπορούν να βρεθούν χρησιμοποιώντας μια αναζήτηση κατά πλάτος (*BFS-Breadth-First Search*) ή αναζήτηση κατά βάθος (*DFS - Depth-first search*). Αυτές είναι δύο από τις πιο σημαντικές τυπικές μεθόδους στην επιστήμη των υπολογιστών. Με τη βοήθειά τους μπορεί κανείς να καθορίσει τα σκούρα γκρι και τα λευκά νούφαρα. Τέλος, πρέπει να βρεθούν μόνο τα νούφαρα, στα οποία μπορούν να φτάσουν και οι δύο βάτραχοι.

Λέξεις κλειδιά και Websites

- αναζήτηση κατά πλάτος (*BFS-Breadth-First Search*):
https://en.wikipedia.org/wiki/Breadth-first_search
Breitensuche: <https://de.wikipedia.org/wiki/Breitensuche>
- αναζήτηση κατά βάθος (*DFS - Depth-first search*):
https://en.wikipedia.org/wiki/Depth-first_search
Tiefensuche: <https://de.wikipedia.org/wiki/Tiefensuche>







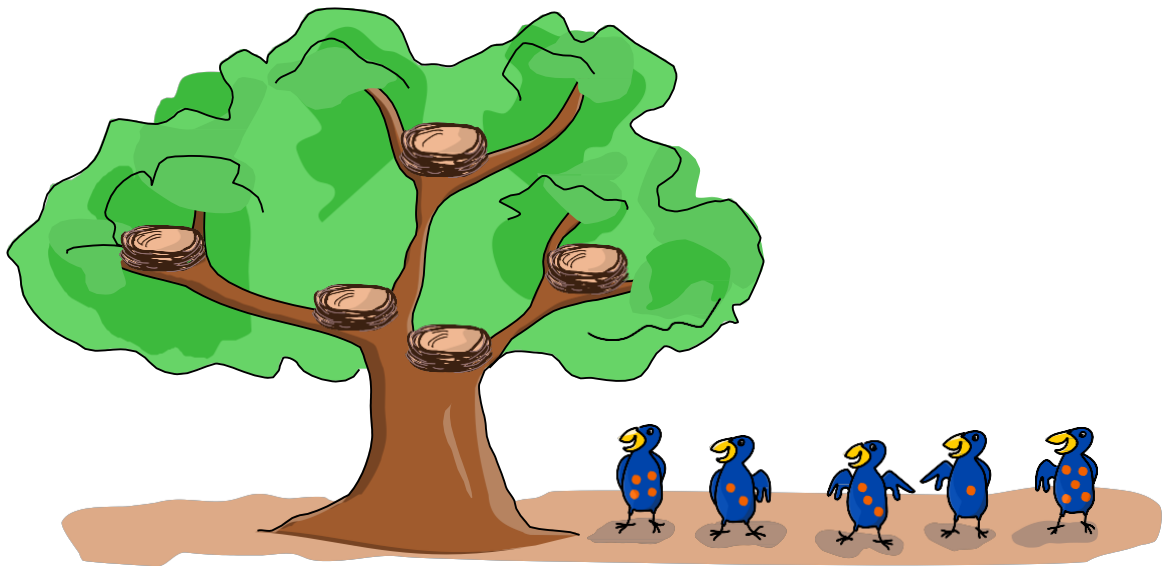
12. Dottis

Τα Dottis είναι πουλιά με τελίτσες στην κοιλιά τους. Δίπλα από ένα δένδρο στέκονται 5 Dottis. Ένα μετά το άλλο -με σειρά από αριστερά προς τα δεξιά- σκαρφαλώνουν στο δένδρο και κινούνται προς στις άδειες φωλιές. Αυτό με τις τέσσερις τελίτσες είναι το πρώτο. Κάθε Dotti προχωρά ως εξής:

Ξεκινά κάτω από το δέντρο. Εκτελεί τα ακόλουθα βήματα μέχρι να βρει μια άδεια φωλιά:

1. Σκαρφαλώνει ψηλά έως ότου βρει μια φωλιά.
2. Εάν η φωλιά είναι άδεια, τότε μπαίνει στη φωλιά και εγκαθίσταται εκεί.
3. Αλλιώς συνεχίζει να σκαρφαλώνει. . .
 - αν το Dotti που κάθεται στη φωλιά έχει περισσότερες τελίτσες τότε σκαρφαλώνει προς τα αριστερά
 - αν το Dotti που κάθεται στη φωλιά έχει ίσες ή λιγότερες τελίτσες σκαρφαλώνει προς τα δεξιά.

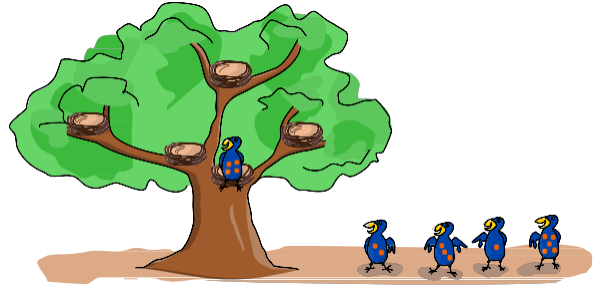
Που θα βρίσκονται τα Dottis στο τέλος; Βάλε το κάθε Dotti στη σωστή φωλιά.



Λύση

Πώς να βρείτε τη σωστή λύση:

Το πρώτο Dotti, αυτό με τις 4 τελίτσες, πηγαίνει στην κάτω κάτω φωλιά και φωλιάζει εκεί.



Το δεύτερο Dotti έχει 2 τελίτσες. Στην κάτω κάτω φωλιά κάθεται το πρώτο Dotti με 4 τελίτσες. Επειδή το $4 > 2$, το δεύτερο Dotti σκαρφαλώνει προς τα αριστερά και καταλαμβάνει την πρώτη ελεύθερη φωλιά.



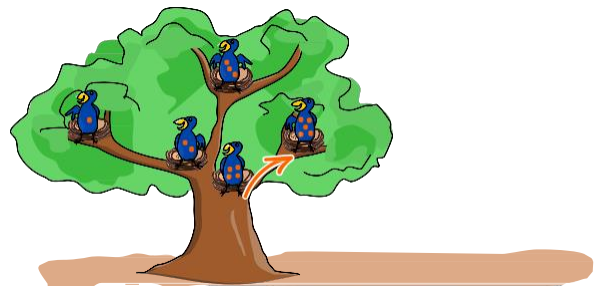
Το τρίτο Dotti έχει 3 τελίτσες. Σκαρφαλώνει στην κάτω κάτω φωλιά όπου κάθεται το Dotti με τις 4 τελίτσες. Επειδή $4 > 3$ συνεχίζει αριστερά. Στην επόμενη φωλιά βρίσκεται το Dotti με τις δύο τελίτσες. Επειδή $2 < 3$, το τρίτο Dotti σκαρφαλώνει προς τα δεξιά και συναντά την επόμενη ελεύθερη φωλιά. Αυτή είναι η φωλιά που βρίσκεται πιο ψηλά από όλες τις άλλες φωλιές.



Το τέταρτο Dotti έχει 1 τελίτσα. Επειδή όλα τα άλλα Dottis έχουν περισσότερες τελίτσες, σε κάθε κατειλημμένη φωλιά που βρίσκει σκαρφαλώνει προς τα αριστερά. Φτάνει έτσι στη φωλιά που βρίσκεται τέρμα αριστερά και φωλιάζει εκεί.



Το τελευταίο Dotti έχει 5 τελίτσες. Επειδή κανένα Dotti δεν έχει περισσότερες τελίτσες, φτάνοντας στις κατειλημμένες φωλιές θα συνεχίζει σκαρφαλώνοντας δεξιά. Αυτό το κάνει μία φορά, μόλις φτάσει στην κάτω κάτω φωλιά σκαρφαλώνει δεξιά και βρίσκει μια άδεια φωλιά.





Αυτό είναι Πληροφορική!

Εάν τα Dottis βρίσκουν φωλιά χρησιμοποιώντας αυτή τη μέθοδο, αυτό έχει ένα ενδιαφέρον πλεονέκτημα: Ένα συγκεκριμένο Dotti μπορεί στη συνέχεια να βρεθεί γρήγορα. Εάν το Dotti που ψάχνετε έχει λιγότερες τελίτσες από αυτό που κοιτάζετε, πρέπει να συνεχίσετε να κοιτάζετε στο αριστερό μέρος του δέντρου. Διαφορετικά, συνεχίστε να κοιτάτε προς τα δεξιά. Έτσι, με κάθε δοκιμή μπορείτε να περιορίσετε την περιοχή αναζήτησης στο μισό. Γι' αυτό θα βρείτε γρήγορα το Dotti σας.

Υπάρχουν πολλοί τρόποι οργάνωσης των δεδομένων, μιλάμε για διαφορετικές *δομές δεδομένων (data structures)*. Η δομή δεδομένων σε αυτήν την εργασία είναι ένα *δυναμικό δέντρο αναζήτησης (binary search tree-BST)*. Η λέξη "δυναμικό" (binary) προέρχεται από τη λατινική λέξη "bis" που σημαίνει "δύο φορές". Γιατί στο τέλος ενός κλαδιού (όπου υπάρχει φωλιά στην άσκηση) υπάρχουν το πολύ δύο μικρότερα κλαδιά προς τα πάνω. Τα δυναμικά δέντρα αναζήτησης χρησιμοποιούνται σε προγράμματα υπολογιστών όταν πρέπει να βρεθούν γρήγορα πολλά δεδομένα. Συνήθως είναι πολύ μεγαλύτερα από το μικρό δέντρο της άσκησης. Υπάρχει επίσης μια άλλη διαφορά: το δέντρο στην άσκηση αφορούσε ένα σταθερό αριθμό πέντε πουλιών. Σε ένα δυναμικό δέντρο αναζήτησης όμως, μπορείτε συνήθως να συνεχίσετε να εισάγετε όλο και περισσότερα δεδομένα. Για να εισαχθεί ένα νέο δεδομένο, ένα νέο κλαδί απλώς προσαρτάται στο άκρο ενός κλαδιού, μεγεθύνοντας έτσι το δέντρο. Οι δομές δεδομένων που μπορούν να αλλάξουν με αυτόν τον τρόπο ονομάζονται *δυναμικές δομές δεδομένων (dynamic data structures)*

Λέξεις κλειδιά και Websites

- δυναμικό δέντρο αναζήτησης (binary search tree-BST):
https://en.wikipedia.org/wiki/Search_tree
Binärer Suchbaum: https://de.wikipedia.org/wiki/Binärer_Suchbaum

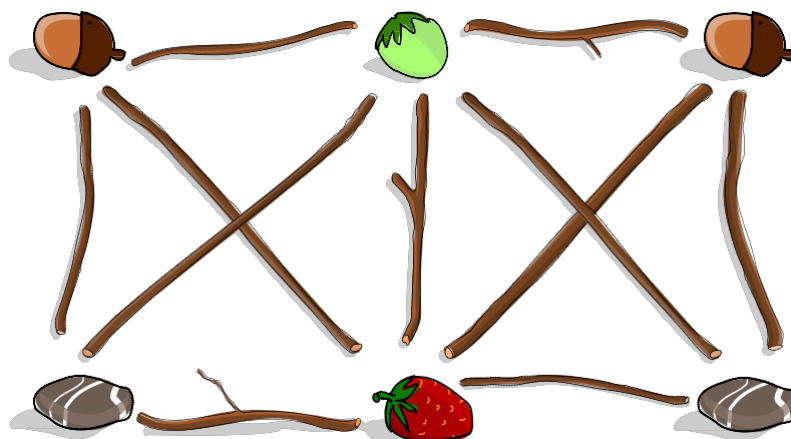




13. Κλέβοντας τη φράουλα

Η Άννα θέλει να δημιουργήσει ένα έργο τέχνης στον κήπο και για αυτό το σκοπό έχει συλλέξει διάφορα πράγματα: Βελανίδια, φουντούκια, πετραδάκια και μια φράουλα. Βάζει μερικά από τα πράγματα στο γκαζόν.

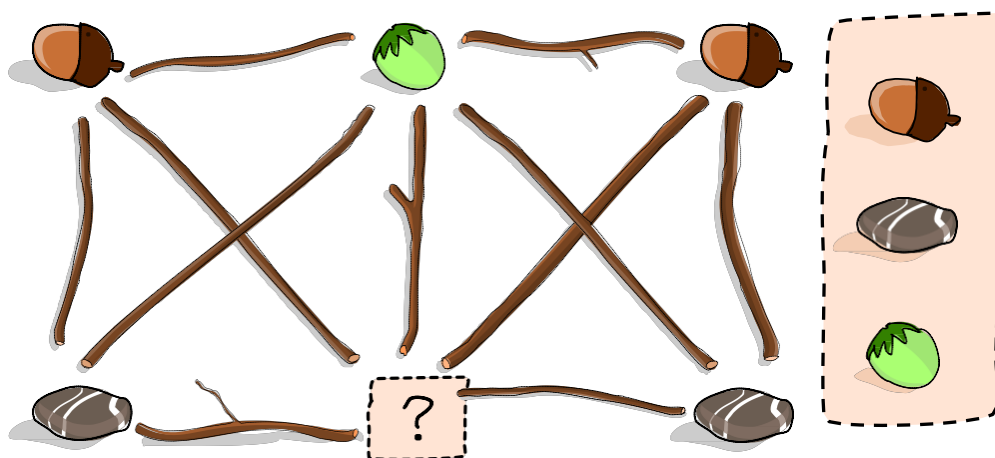
Τότε η Άννα βάζει κλαδιά ανάμεσα σε αυτά τα πράγματα. Με αυτόν τον τρόπο, ακολουθεί τον ακόλουθο κανόνα: Ένα κλαδί δεν πρέπει να βρίσκεται ανάμεσα σε δύο ίδια αντικείμενα – για παράδειγμα, όχι ανάμεσα σε δύο βελανίδια. Εδώ είναι το ολοκληρωμένο έργο τέχνης:



Ενώ η Άννα φεύγει, έρχεται ο αδελφός της και τρώει τη φράουλα.

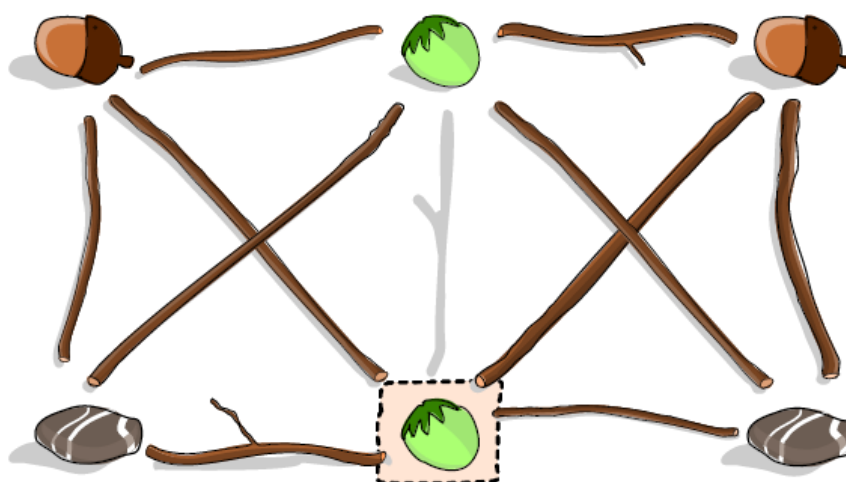
Μπορείτε να τον βοηθήσετε να καλύψει τη ζαβολιά;

Τοποθετήστε ένα άλλο πράγμα στη θέση της φράουλας και αφαιρέστε ακριβώς ένα κλαδί. Στο τέλος πρέπει ο κανόνας της Άννας να ισχύει επίσης για το τροποποιημένο έργο τέχνης.



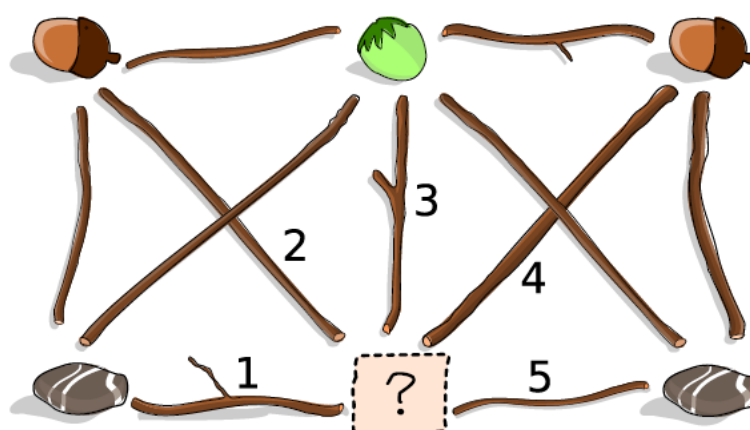
Λύση

Εάν αντικαταστήσετε τη φράουλα με ένα φουντούκι, το κλαδί 3 στη μέση παραβιάζει τον κανόνα της Άννας: βρίσκεται ανάμεσα σε δύο πανομοιότυπα πράγματα, δηλαδή σε δύο φουντούκια. Επομένως, αυτό το κλαδί πρέπει να αφαιρεθεί.



Οι άλλες δύο πιθανές αντικαταστάσεις απαιτούν την αφαίρεση περισσότερων από ένα κλαδιών:

- Εάν η φράουλα αντικατασταθεί από βελανίδι, πρέπει να αφαιρέσετε τα κλαδιά 2 και 4.
- Εάν η φράουλα αντικατασταθεί από πετραδάκι, πρέπει να αφαιρέσετε τα κλαδιά 1 και 5.



Αυτό είναι Πληροφορική!

Το έργο τέχνης της Άννας μπορεί να αναπαρασταθεί ως γράφος (*graph*). Ένας γράφος αποτελείται από *κόμβους* (*nodes*) (τις θέσεις για τα πράγματα) και *ακμές* (*edges*) (τα κλαδιά), που συνδέουν δύο κόμβους μεταξύ τους. Οι γράφοι έχουν πολλές χρήσεις και χρησιμοποιούνται για μοντελοποίηση σε πολλά προβλήματα της επιστήμης των υπολογιστών. Όταν δύο κόμβοι συνδέονται απευθείας με μια ακμή, λέγονται *γειτονικοί*. Μια ομάδα κόμβων στους οποίους κάθε κόμβος είναι γείτονας κάθε άλλου κόμβου ονομάζεται *κλίκα* (*clique*). Στο γράφημά μας έχουμε δύο κλίκες με τέσσερις κόμβους: το δεξί και το αριστερό μισό του γραφήματος. (Το φουντούκι πάνω και το ερωτηματικό ανήκουν και στις δύο κλίκες.) Από τον κανόνα της Άννας προκύπτει ότι σε όλους τους κόμβους μιας κλίκας πρέπει να υπάρχουν διαφορετικά πράγματα. Για να είμαστε σύμφωνοι με τον κανόνα χρειαζόμαστε τουλάχιστον τόσα διαφορετικά πράγματα όσοι είναι οι κόμβοι μιας κλίκας. Αφού αφαιρέσουμε τη φράουλα, έχουμε μόνο 3 διαφορετικά είδη. Έτσι, τώρα μπορούν να παραμείνουν κλίκες που έχουν το μέγιστο 3 κόμβους, εάν θέλουμε ο κανόνας να συνεχίσει να εκπληρώνεται. Πρέπει λοιπόν να αφαιρεθεί μια ακμή (ένα κλαδί), σπάζοντας και τις δύο κλίκες με τέσσερις κόμβους.

Ο κανόνας της Άννας αντιστοιχεί σε έναν κανόνα στο λεγόμενο *πρόβλημα χρωματισμού των γράφων* (*graph-coloring problem*): εκχωρούμε ένα χρώμα σε κάθε κόμβο ενός γραφήματος, όπου οι γείτονες πρέπει να έχουν διαφορετικά χρώματα. (Τα χρώματα αντιστοιχούν σε διαφορετικούς τύπους πραγμάτων.) Ο στόχος είναι συνήθως να χρησιμοποιηθούν όσο το δυνατόν λιγότερα χρώματα. Το πρόβλημα του τρόπου χρωματισμού ενός γράφου με τον ελάχιστο αριθμό χρωμάτων έχει πολλές εφαρμογές. Μερικά παραδείγματα είναι ο προγραμματισμός αθλητικών αγώνων, ο σχεδιασμός ενός πίνακα θέσεων ακόμη και η επίλυση ενός Sudoku.

Λέξεις κλειδιά και Websites

- πρόβλημα χρωματισμού των γράφων (*graph-coloring problem*):
https://en.wikipedia.org/wiki/Graph_coloring
Färbeproblem: [https://de.wikipedia.org/wiki/Färbung_\(Graphentheorie\)](https://de.wikipedia.org/wiki/Färbung_(Graphentheorie))
- χρωματισμός ακμών (*edge-coloring*), ελάχιστος χρωματισμός ακμών (*minimal edge coloring*), https://en.wikipedia.org/wiki/Edge_coloring
minimale Kantenfärbung, *Kantenfärbung*: <https://de.wikipedia.org/wiki/Kantenfärbung>
- κλίκα (*clique*): [https://en.wikipedia.org/wiki/Clique_\(graph_theory\)](https://en.wikipedia.org/wiki/Clique_(graph_theory))
[https://de.wikipedia.org/wiki/Clique_\(Graphentheorie\)](https://de.wikipedia.org/wiki/Clique_(Graphentheorie))

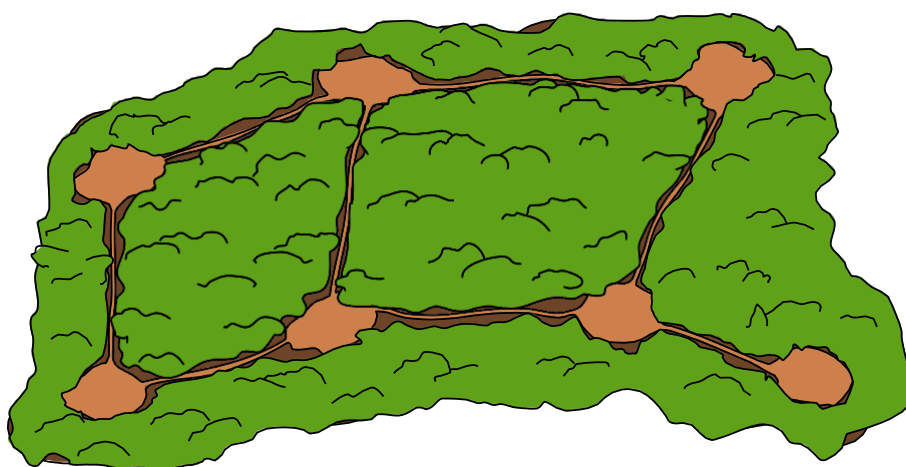




14. Παρατηρώντας τα ζώα

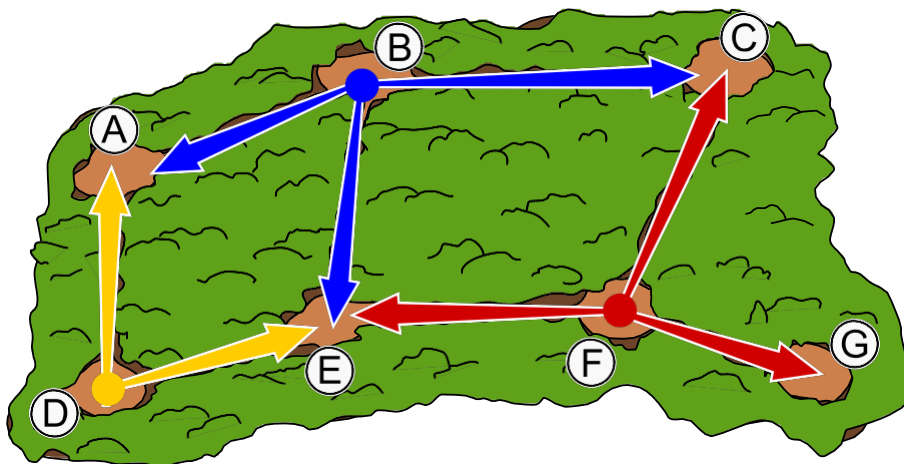
Οι δασοφύλακες θέλουν να παρατηρούν τα ζώα στα μονοπάτια του δάσους. Από κάθε ξέφωτο μπορούν να παρατηρήσουν όλες τις εξερχόμενες διαδρομές μέχρι το επόμενο ξέφωτο. Όσο το δυνατόν λιγότεροι δασοφύλακες θα πρέπει να παρατηρούν όλα τα μονοπάτια.

Επιλέξτε όσο το δυνατόν λιγότερα ξέφωτα από τα οποία οι δασοφύλακες μπορούν να παρατηρήσουν όλα τα μονοπάτια!



Λύση

Η εικόνα δείχνει την «ελάχιστη» λύση, όπου οι δασολόγοι μπορούν να σταθούν μόνο σε τρία ξέφωτα και να παρατηρούν όλα τα μονοπάτια.



Υπάρχουν οκτώ μονοπάτια για παρακολούθηση. Εάν μόνο δύο δασοφύλακες μπορούσαν να παρατηρήσουν όλα τα μονοπάτια, θα έπρεπε να υπάρχει ένα ξέφωτο από το οποίο φεύγουν τουλάχιστον τέσσερα μονοπάτια. Αλλά δεν υπάρχει τέτοιο ξέφωτο σε αυτό το δάσος. Επομένως, δύο δασολόγοι δεν αρκούν.

Ως εκ τούτου, απαιτούνται τουλάχιστον τρεις δασολόγοι για την παρακολούθηση όλων των μονοπατιών. Επομένως, η λύση που δίνεται εδώ είναι μια λύση με τον μικρότερο δυνατό αριθμό δασολόγων. Πραγματικά δεν υπάρχει άλλη λύση με ακριβώς τρεις δασολόγους.

Από τον αριθμό των μονοπατιών που πρέπει να παρατηρηθούν και το γεγονός ότι δεν υπάρχουν ξέφωτα με περισσότερα από τρία μονοπάτια, μπορούμε να συμπεράνουμε: Κάθε δασολόγος πρέπει να παρατηρεί τουλάχιστον δύο μονοπάτια που δεν παρατηρούν οι άλλοι δασολόγοι.

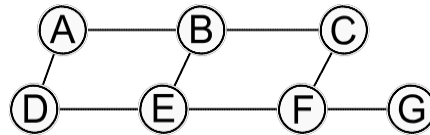
Για να ξεπεραστεί το αδιέξοδο μεταξύ του ξέφωτου F και G, πρέπει να τοποθετηθεί ένας δασολόγος στο F. Προκειμένου να παρατηρείται το μονοπάτι μεταξύ B και C, ο δεύτερος δασοφύλακας πρέπει να παρατηρεί από το ξέφωτο B. Για να παρατηρούνται τα δύο τελευταία μονοπάτια με έναν μόνο δασολόγο, πρέπει να τοποθετηθεί ένας στο ξέφωτο D. Αυτό σημαίνει ότι η λύση που δίνεται είναι υποχρεωτική και δεν μπορεί να υπάρξει άλλη.





Αυτό είναι Πληροφορική!

Οι σχέσεις μεταξύ αντικειμένων (π.χ. μονοπάτια μεταξύ ξέφωτων στο δάσος) μπορούν να αναπαρασταθούν ως γράφος (*graph*). Ένας γράφος αποτελείται από κόμβους (*nodes*) (εδώ: τα ξέφωτα), που αναπαρίστανται ως κύκλοι, και ακμές (*edges*) (εδώ: τα μονοπάτια), που αναπαρίστανται ως γραμμές μεταξύ των κόμβων. Ο γράφος αυτής της άσκησης μοιάζει με αυτό:



Σε αυτή την άσκηση θα πρέπει πει να βρείτε τον μικρότερο αριθμό κόμβων στο γράφημα έτσι ώστε κάθε ακμή να ξεκινά ή να τελειώνει σε τουλάχιστον έναν από αυτούς τους κόμβους. Οι επιστήμονες υπολογιστών αποκαλούν ένα τέτοιο υποσύνολο των κόμβων ως *ελάχιστη κάλυψη κορυφών* (*minimal vertex cover*). Τέτοια προβλήματα βρίσκουμε συχνά στην καθημερινή ζωή, για παράδειγμα όταν αναζητούμε τα καλύτερα σημεία για λαμπτήρες δρόμου ή για την έξυπνη τοποθέτηση καμερών παρακολούθησης.

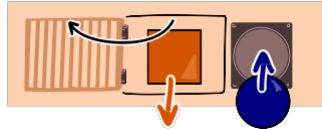
Λέξεις κλειδιά και Websites

- γράφος (*graph*): https://en.wikipedia.org/wiki/Graph_theory
Graph: [https://de.wikipedia.org/wiki/Graph_\(Graphentheorie\)](https://de.wikipedia.org/wiki/Graph_(Graphentheorie))
- πρόβλημα κάλυψης κορυφών (*vertex cover problem*, *node cover problem*):
https://en.wikipedia.org/wiki/Vertex_cover
Knotenüberdeckung: <https://de.wikipedia.org/wiki/Knotenüberdeckung/>



15. Αίνιγμα στα γενέθλια

Ο Σεβαστιανός παίρνει ένα κουτί με 15 πόρτες για τα γενέθλιά του. Πίσω από τη μεσαία πόρτα είναι ένα επιπλέον δώρο. Πίσω από τις άλλες πόρτες υπάρχουν πλακάκια με διαφορετικά σχήματα. Κάθε πόρτα έχει μια τρύπα, στα δεξιά της. Ο Σεβαστιανός μπορεί να ανοίξει μια πόρτα βάζοντας ένα πλακάκι του ίδιου σχήματος στην τρύπα - σαν ένα κλειδί.



Στην αρχή ο Σεβαστιανός έχει αυτό το στρογγυλό πλακάκι: 

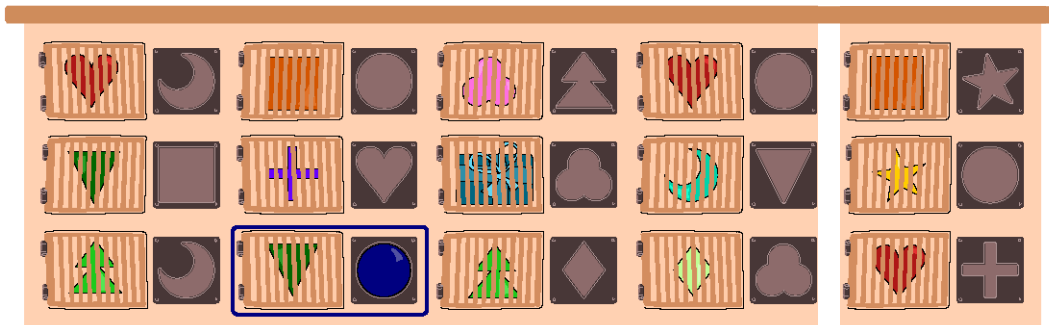
Θέλει να ανοίξει το πολύ πέντε πόρτες για να φτάσει στο δώρο.

Ποια πόρτα πρέπει να ανοίξει πρώτη ο Σεβαστιανός;

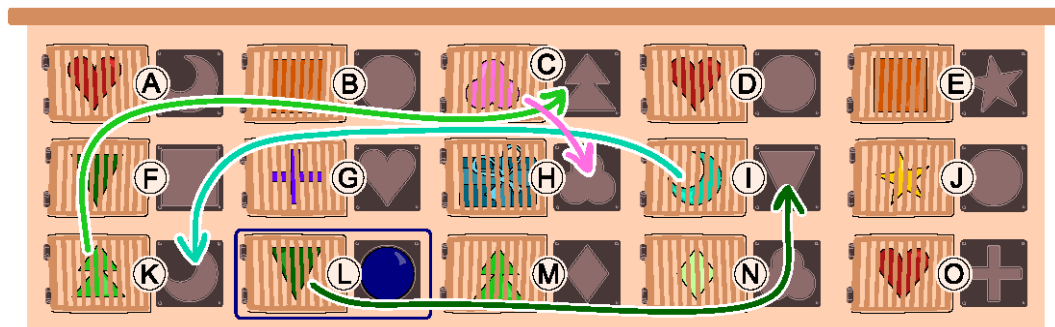


Λύση

Ο Σεβαστιανός πρέπει πρώτα να ανοίξει την πόρτα που έχουμε σημειώσει με μπλε χρώμα:



Στην παρακάτω εικόνα, οι πόρτες έχουν αριθμούς και τα βέλη δείχνουν πώς ο Σεβαστιανός φθάνει στο δώρο με συνολικά 5 ανοίγματα πορτών.



Μπορούμε να παρουσιάσουμε τη σειρά με την οποία ανοίγει τις 5 πόρτες και ως εξής:



Θα υπήρχαν επίσης και άλλες διαδρομές για να φτάσουμε στο δώρο, για παράδειγμα η παρακάτω.



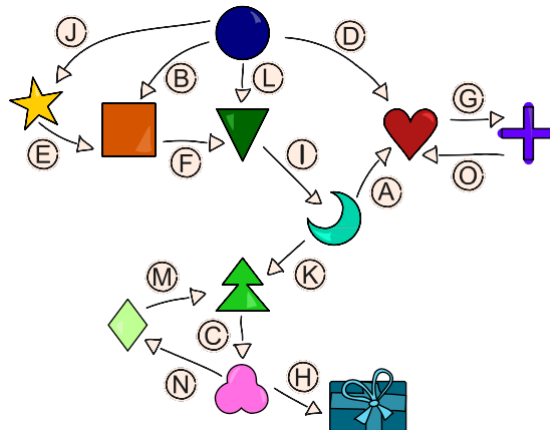
Αλλά αυτές οι διαδρομές είναι πάρα πολύ μακριές, θα έπρεπε να ανοίξουν περισσότερες από πέντε πόρτες. Η δοκιμή όλων των δυνατοτήτων είναι αρκετά χρονοβόρα.

Στην προκειμένη περίπτωση, βρίσκει κανείς τον συντομότερο δρόμο και συνεπώς τη σωστή λύση πιο γρήγορα με μια λεγόμενη αναζήτηση προς τα πίσω: ξεκινάει από την πόρτα με το δώρο και μετά εξετάζει ποιο πλακάκι κάθε φορά χρειάζεται.



Αυτό είναι Πληροφορική!

Με περισσότερο χρόνο και προσπάθεια, μπορούμε επίσης να αναπαραστήσουμε την κατάσταση που έχουμε στην άσκηση μας ως γράφο (graph):



Ένας γράφος αποτελείται γενικά από κόμβους (nodes) (κύκλους) και ακμές (edges) (γραμμές) μεταξύ των κόμβων. Στην περίπτωση μας έχουμε έναν κόμβο για κάθε σχήμα και έναν κόμβο με το δώρο. Οι ακμές εδώ είναι βέλη (ονομάζονται επίσης κατευθυνόμενες ακμές, directed edges) και αντιστοιχούν στις πόρτες. Κάθε βέλος οδηγεί από το σχήμα για να ανοίξει την πόρτα στο σχήμα που βρίσκεται πίσω από την πόρτα.

Η επιστήμη των υπολογιστών εργάζεται πολύ με γραφήματα. Από τη μια πλευρά, συχνά προσφέρουν παραστατικές αναπαραστάσεις αφηρημένων σχέσεων.

Από την άλλη πλευρά, υπάρχουν έτοιμοι αλγόριθμοι που απαντούν πολύ αποτελεσματικά στις ερωτήσεις που σχετίζονται με γράφους. Για πιο περίπλοκες εργασίες, η προσπάθεια που απαιτείται για την κατασκευή του γράφου αξίζει τον κόπο.

Στη συγκεκριμένη άσκηση ψάχνουμε μια διαδρομή μέγιστου μήκους 5, από το πλακάκι που παίρνουμε αρχικά  στο δώρο .

Ένας καλός αλγόριθμος για αυτή την περίπτωση θα ήταν ο αλγόριθμος αναζήτησης κατά πλάτος (BFS-Breadth-First Search algorithm). Αυτός μπορεί να εφαρμοστεί τόσο για γράφους με κατευθυνόμενες ακμές, όπως στην άσκηση, όσο και για γράφους με μη-κατευθυνόμενες ακμές.

Λέξεις κλειδιά και Websites

- κατευθυνόμενος γράφος (directed graph): https://en.wikipedia.org/wiki/Directed_graph
Gerichteter Graph: https://de.wikipedia.org/wiki/Gerichteter_Graph
- αναζήτηση κατά πλάτος (BFS-Breadth-First Search):
https://en.wikipedia.org/wiki/Breadth-first_search
Breitensuche: <https://de.wikipedia.org/wiki/Breitensuche>







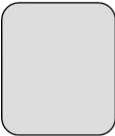
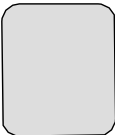
16. Το αγαπημένο δώρο

Η καστορο-οικογένεια έχει τρία δώρα για τα τρία παιδιά της. Κάθε παιδί λέει πρώτο το πιο αγαπημένο του δώρο και αμέσως μετά το δεύτερο πιο αγαπημένο. Τα δώρα πρέπει να μοιραστούν με τον σωστό τρόπο:

1. Όσο το δυνατόν πιο πολλά παιδιά πρέπει να πάρουν ως δώρο την πρώτη τους προτίμηση.

2. Αν κάποιο παιδί δεν πάρει το πιο αγαπημένο δώρο (την πρώτη προτίμηση του), θα πάρει το δεύτερο πιο αγαπημένο του δώρο (την δεύτερη προτίμηση του)

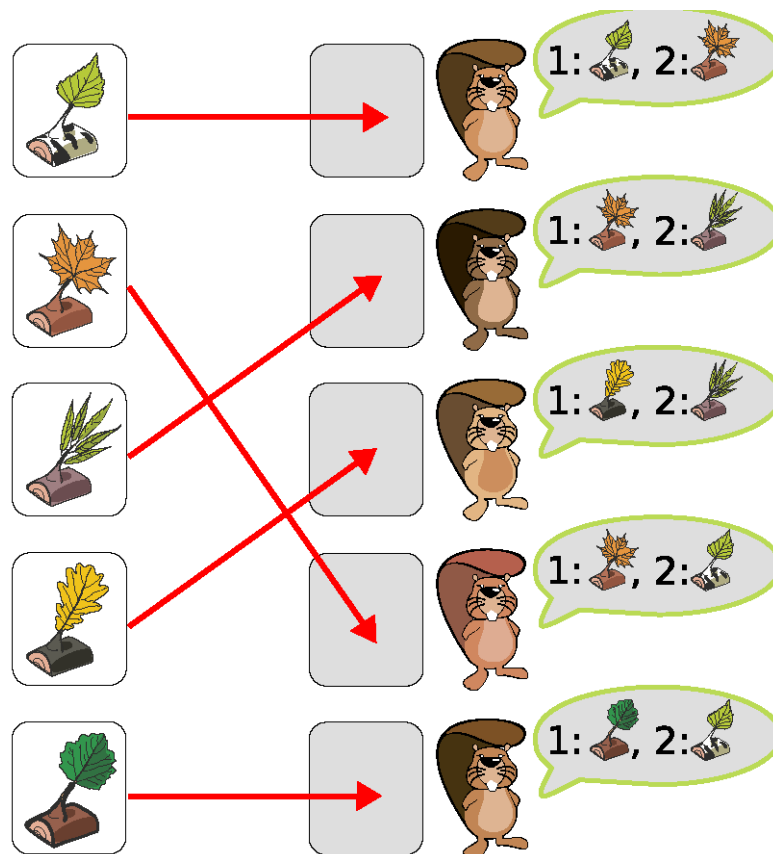
Δώσε στα παιδιά το σωστό δώρο.

		 1:  , 2: 
		 1:  , 2: 
		 1:  , 2: 
		 1:  , 2: 
		 1:  , 2: 



Λύση

Εδώ είναι η μοναδική αντιστοίχιση των δώρων που πληροί και τις δύο συνθήκες.



Στο παραπάνω γράφημα βλέπουμε ότι 4 παιδιά πήραν το αγαπημένο τους δώρο και ένα το δεύτερο αγαπημένο του. Δεν μπορούν όλα τα παιδιά να πάρουν το πιο αγαπημένο τους δώρο (την πρώτη τους προτίμηση), καθώς δύο έχουν το ίδιο πιο αγαπημένο δώρο. Άρα δεν είναι δυνατή μια μοιρασιά όπου όλοι θα πάρουν το πιο αγαπημένο τους δώρο. Σημείωση: Εάν κάνετε την κατανομή με τη σειρά από πάνω προς τα κάτω και διαθέσετε το δεύτερο πιο αγαπημένο δώρο στον δεύτερο κάστορα, τότε ο τέταρτος κάστορας δεν θα λάβει κανένα από τα δώρα που προτιμά. Σε αυτήν την άσκηση δεν αρκεί να κάνετε την καλύτερη εκ πρώτης όψεως διαθέσιμη επιλογή για κάθε κάστορα.

Μια στρατηγική λύσης είναι να ταιριάζετε πρώτα όλα τα δώρα που είναι πρώτη προτίμηση μόνο από ένα παιδί. Μετά από αυτό, μόνο δύο παιδιά παραμένουν με το ίδιο αγαπημένο δώρο. Με αυτά βλέπετε σε ποιον μπορεί να ανατεθεί το δεύτερο αγαπημένο δώρο. Δίνεις στον άλλον το αγαπημένο του δώρο





Αυτό είναι Πληροφορική!

Αυτή η άσκηση είναι ένα πρόβλημα μοναδικής αντιστοίχισης (*matching*): θέλουμε να αντιστοιχίσουμε τα δώρα έτσι ώστε όλα τα παιδιά να πάρουν ένα δώρο και να μην υπάρχει παιδί χωρίς δώρο. Τα παιδιά δεν έχουν μόνο μια προτίμηση, αλλά δίνουν μια σειρά προτιμήσεων. Τέτοια προβλήματα αντιστοίχισης με διαβαθμισμένες προτιμήσεις (1^η προτίμηση, 2^η προτίμηση, κοκ) μπορεί να γίνουν πολύ περίπλοκα. Η επιστήμη των υπολογιστών μας βοηθά να λύσουμε τέτοια προβλήματα όσο το δυνατόν πιο γρήγορα.

Μια δυνατότητα είναι να δώσουμε στις αντιστοιχίσεις μια τιμή: το αγαπημένο δώρο έχει τιμή 1 και το δεύτερο αγαπημένο δώρο έχει τιμή 2. Θέλουμε να ελαχιστοποιήσουμε τη συνολική τιμή. Μια αντιστοίχιση είναι η βέλτιστη (*optimal*) εάν δεν υπάρχει άλλη αντιστοίχιση με περισσότερες εκπληρωμένες πρώτες αντιστοιχίσεις. Στην επιστήμη των υπολογιστών, μια τέτοια εργασία αναφέρεται ως *Rank-Maximal-Ranking*. Υπάρχουν πολλά προβλήματα αντιστοίχισης (*Matching-Problems*). Ένα από αυτά ονομάζεται *πρόβλημα του σταθερού παντρέματος* (engl. *Stable Marriage Problem*). Ακούγεται ενδιαφέρον? Η επιστήμη των υπολογιστών είναι ένα πολύπλευρο επιστημονικό πεδίο.

Λέξεις κλειδιά και Websites

- Πρόβλημα αντιστοίχισης (Matching problem):
[https://en.wikipedia.org/wiki/Matching_\(graph_theory\)](https://en.wikipedia.org/wiki/Matching_(graph_theory))
Zuordnungsproblem: <https://de.wikipedia.org/wiki/Zuordnungsproblem>
- Rank-maximal matching: https://en.wikipedia.org/wiki/Rank-maximal_allocation
https://en.wikipedia.org/wiki/Ranked_voting
Rang-Wahl: <https://de.wikipedia.org/wiki/Rang-Wahl>



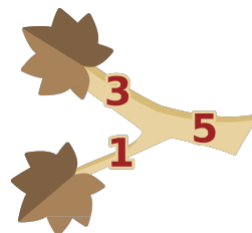


17. Σώσε το δέντρο!

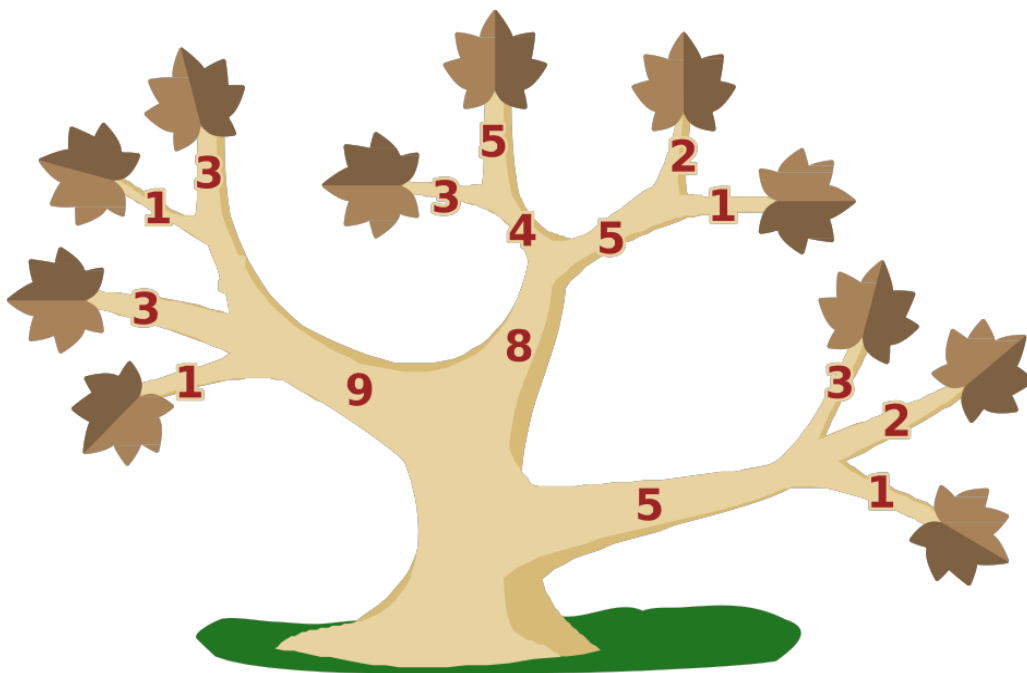
Ένα δέντρο στον κήπο του Μπρούνο είναι άρρωστο, όλα τα φύλλα έχουν ξεραθεί. Ο Μπρούνο θέλει να σώσει το δέντρο. Για να γίνει αυτό, πρέπει να κόψει μερικά κλαδιά, ώστε στο τέλος να αφαιρεθούν όλα τα φύλλα. Τότε μπορούν να αναπτυχθούν νέα κλαδιά με νέα φύλλα. Ο Μπρούνο θέλει να τελειώσει το συντομότερο δυνατό. Οι αριθμοί στα κλαδιά δείχνουν το χρόνο για την ολοκλήρωση του κοψίματος:

Η εικόνα δείχνει ένα παράδειγμα:

Για να απομακρύνει τα δύο φύλλα, ο Μπρούνο μπορεί είτε να κόψει ξεχωριστά τα δύο κλαδιά, είτε το κλαδί από το οποίο ξεκινούν τα δύο κλαδιά. Οι αριθμοί δείχνουν το χρόνο που διαρκεί το κόψιμο του κάθε κλαδιού. Ο Μπρούνο θα επιλέξει να κόψει τα δύο κλαδιά με τα φύλλα, διότι $3 + 1 < 5$. Παρακάτω φαίνεται όλο το δέντρο.

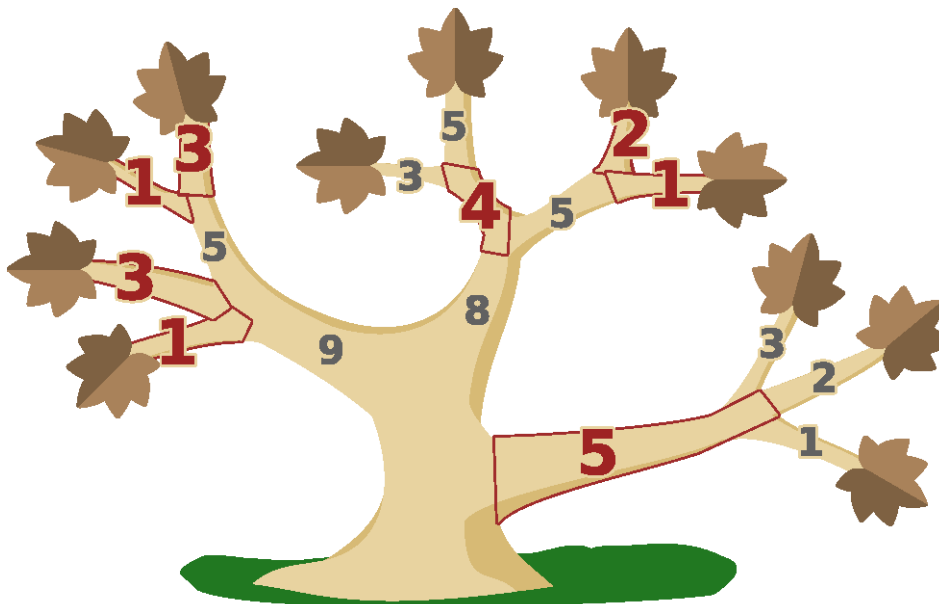


Ποια κλαδιά θα πριονίσει ο Μπρούνο έτσι ώστε να τελειώσει όσο γίνεται πιο γρήγορα ;



Λύση

Αυτή είναι η λύση: Ο Μπρούνο πριονίζει τα κλαδιά που έχουν σημειωθεί με κόκκινο χρώμα έτσι ώστε να τελειώσει όσο πιο γρήγορα γίνεται:



Αλλά γιατί είναι έτσι; Πρώτα μπορούμε να υπολογίσουμε πόσο χρόνο χρειάζεται ο Μπρούνο εάν έκοβε τα κλαδιά που έχουν φύλλα (τα «τελικά» κλαδιά), τότε θα είχε ολοκληρώσει το έργο του:

$$1+3+1+3+3+5+2+1+3+2+1=25$$

Τώρα συνεχίζουμε προς την κατεύθυνση του κορμού και συνεχίζουμε να σκεφτόμαστε αν θα ήταν πιο γρήγορο να κόψουμε το κλαδί από το οποίο διακλαδίζονται άμεσα ή έμμεσα τα προηγούμενα κλαδιά. Μετά το πρώτο τέτοιο βήμα, προκύπτει ο ακόλουθος υπολογισμός (η συνάρτηση «min» υπολογίζει το ελάχιστο των ορισμάτων της):

$$\begin{aligned} &1 + 3 + \min(5, 1 + 3) + \min(4, 3 + 5) + \min(5, 2 + 1) + \min(5, 3 + 2 + 1) \\ &= 1 + 3 + (1 + 3) + 4 + (2 + 1) + 5 \\ &= 20 \end{aligned}$$

Εδώ δεν υπολογίζουμε τον συνολικό χρόνο που χρειαζόμαστε για όλο το δέντρο, αλλά βλέπουμε ποια μητρικά κλαδιά (κλαδιά από τα οποία ξεκινούν άλλα κλαδιά) πρέπει να κοπούν. Στο επόμενο βήμα έχουμε ήδη φτάσει στον κορμό:

$$\begin{aligned} &\min(9, 1 + 3 + 1 + 3) + \min(8, 4 + 2 + 1) + 5 \\ &= (1 + 3 + 1 + 3) + (4 + 2 + 1) + 5 \\ &= 20 \end{aligned}$$

Πιο γρήγορα δεν θα μπορούσε να κάνει ο Μπρούνο.



Αυτό είναι πληροφορική!

Ας υποθέσουμε ότι τα κομμένα κομμάτια του δέντρου του Μπρούνο δεν πέφτουν κατευθείαν στο έδαφος. Τότε θα μπορούσαμε να πούμε ότι το δέντρο χωρίζεται σε δύο μέρη: το ένα μέρος περιέχει όλα τα κομμάτια που έχουν κοπεί, συγκεκριμένα όλα τα φύλλα, και το άλλο μέρος περιέχει τον κορμό και όλα τα κλαδιά που εκτείνονται από αυτόν μέχρι τα σημεία που έγινε η κοπή. Αυτή η διαίρεση ή η *κοπή (cut)* του δέντρου είναι η ελάχιστη σε σχέση με τον χρόνο που πρέπει να αφιερώσει ο Μπρούνο για να το κόψει.

Η επιστήμη των υπολογιστών γνωρίζει επίσης τα δέντρα (*trees*) και τα χρησιμοποιεί για να αναπαραστήσει αντικείμενα που συνδέονται μεταξύ τους με συγκεκριμένο τρόπο. Τα αντικείμενα ονομάζονται *κόμβοι*, οι συνδέσεις ονομάζονται *ακμές*. Υπάρχει πάντα μόνο ένα μονοπάτι μεταξύ δύο κόμβων - ακριβώς όπως σε ένα πραγματικό δέντρο υπάρχει πάντα μόνο ένα μονοπάτι κατά μήκος των κλαδιών προς τον κορμό από ένα φύλλο ή μια διχάλα κλαδιού. Αν κάποιος παραλείψει αυτή τη συνθήκη, μιλάει γενικότερα για ένα γράφο (*graph*).

Σε ένα γράφο γενικά, μια *ελάχιστη τομή (minimal cut)*, δηλαδή η διαίρεση σε δύο ή περισσότερα μέρη με ελάχιστο κόστος, δεν είναι τόσο εύκολο να υπολογιστεί όσο δείξαμε εδώ για ένα δέντρο, αλλά δεν είναι και πολύ δύσκολο. Αυτό είναι χρήσιμο, γιατί υπάρχουν ενδιαφέρουσες εφαρμογές. Οι ελάχιστες τομές βρίσκουν, για παράδειγμα, εφαρμογή, κατά τη διαίρεση των αρχείων εικόνας σε όμοια μέρη. Σε ειδικούς γράφους, τα *δίκτυα ροής (flow networks)*, τα οποία, μεταξύ άλλων, μπορούν να χρησιμοποιηθούν για τη μοντελοποίηση των ροών δεδομένων μέσω δικτύων υπολογιστών, το κόστος μιας ελάχιστης τομής αντιστοιχεί στη μέγιστη δυνατή ροή σε ολόκληρο το δίκτυο (βλ. και θεώρημα μέγιστης ροής-ελάχιστης τομής, *Max-Flow-Min-Cut-Theorem*).

Stichwörter und Webseiten

- Δέντρα (*trees*): [https://en.wikipedia.org/wiki/Tree_\(graph_theory\)](https://en.wikipedia.org/wiki/Tree_(graph_theory))
Baum: [https://de.wikipedia.org/wiki/Baum_\(Graphentheorie\)](https://de.wikipedia.org/wiki/Baum_(Graphentheorie))
- Τομή (*cut*): [https://en.wikipedia.org/wiki/Cut_\(graph_theory\)](https://en.wikipedia.org/wiki/Cut_(graph_theory))
Schnitt: [https://de.wikipedia.org/wiki/Schnitt_\(Graphentheorie\)](https://de.wikipedia.org/wiki/Schnitt_(Graphentheorie))
- Flows, cuts in networks (Flüsse und Schnitte in Netzwerken):
https://de.wikipedia.org/wiki/Flüsse_und_Schnitte_in_Netzwerken
- θεώρημα μέγιστης ροής-ελάχιστης τομής (Max-Flow-Min-Cut-Theorem):
<https://de.wikipedia.org/wiki/Max-Flow-Min-Cut-Theorem>





18. Βιβλιοθήκη

Η Σούζυ είναι με τον Τίμο στη βιβλιοθήκη. Θέλουν να δανειστούν το βιβλίο με τίτλο: «Dämme bauen, aber gern!» Ο Τίμος πηγαίνει στη βιβλιοθήκη με το νούμερο 1, κοιτά στο ράφι 3, στο υπο-ράφι 6 και παίρνει το βιβλίο. Η Σούζυ μένει έκπληκτη! Ο Τίμος εξηγεί στη Σούζυ πως βρίσκει τη θέση του βιβλίου:

Παίρνουμε το πρώτο γράμμα κάθε λέξης του τίτλου και προσδιορίζουμε τη θέση του στο αλφάβητο. Σταδιακά, αυτές οι θέσεις προστίθενται, αλλά πριν από κάθε πρόσθεση, η τιμή που έχει επιτευχθεί μέχρι τώρα πολλαπλασιάζεται επί 3. Για το συγκεκριμένο βιβλίο, το αποτέλεσμα είναι 136. Είναι ήδη σαφές πού βρίσκεται το βιβλίο.

a	b	c	d	e	f	g	h	i	j	k	l	m
1	2	3	4	5	6	7	8	9	10	11	12	13
n	o	p	q	r	s	t	u	v	w	x	y	z
14	15	16	17	18	19	20	21	22	23	24	25	26

Dämme bauen, aber gern!

((4 · 3 + 2) · 3 + 1) · 3 + 7

Τώρα η Σούζι κάνει τους αντίστοιχους υπολογισμούς για τα αγαπημένα της βιβλία. Σε μια περίπτωση όμως έκανε λάθος.

Σε ποια?

- | | | | |
|----|---|----|--|
| A) | <p>Gutes gegen Blber-Fleber</p> $((7 \cdot 3 + 7) \cdot 3 + 2) \cdot 3 + 6$ | B) | <p>Bäume fallen für Dummles</p> $((2 \cdot 3 + 6) + 6) \cdot 3 + 4$ |
| C) | <p>Der Herr der Dämme</p> $((4 \cdot 3 + 8) \cdot 3 + 4) \cdot 3 + 4$ | D) | <p>Bebrett! Der erste Fall</p> $((2 \cdot 3 + 4) \cdot 3 + 5) \cdot 3 + 6$ |





Λύση

Η Σούζυ έκανε σχεδόν τα πάντα σωστά: πάντα πρόσθεσε τις σωστές τιμές θέσης και τα ενδιάμεσα αποτελέσματα τα πολλαπλασίασε με το 3 – με μία εξαίρεση: Στην απάντηση Β ξέχασε να το κάνει σε μια περίπτωση.

Bäume fallen für Dummles $((2 \cdot 3 + 6) \cdot 3 + 6) \cdot 3 + 4$

Αυτό είναι Πληροφορική!

Η βιβλιοθήκη δίνει τη δυνατότητα στους επισκέπτες της να προσδιορίσουν την ακριβή θέση των βιβλίων με τις «εκφράσεις προσδιορισμού τόπου». Οπότε κανείς δεν χρειάζεται να ψάξει πολύ. Ωστόσο, η βιβλιοθήκη και οι επισκέπτες πρέπει να έχουν κατά νου ένα πράγμα: οι εκφράσεις και επομένως τα αποτελέσματα μπορεί να είναι τα ίδια για διαφορετικά βιβλία. Για παράδειγμα, το "Bäume fallen für Dummies" και το "Biber finden Fichten dufte" είναι στο ίδιο υπο-ράφι. Τα υπο-ράφια δεν πρέπει να είναι πολύ μικρά ή πρέπει να μπορούν να ρυθμίζονται με ευελιξία.

Ακόμη και με δεδομένα που είναι αποθηκευμένα στη μνήμη του υπολογιστή, είναι καλή ιδέα εάν η θέση τους μπορεί να υπολογιστεί απευθείας από τα ίδια τα δεδομένα. Για το σκοπό αυτό στην επιστήμη των υπολογιστών αναπτύχθηκαν *συναρτήσεις κατακερματισμού (hash-functions)*: μαθηματικές συναρτήσεις που υπολογίζουν μια τιμή από το περιεχόμενο των δεδομένων ή από μέρος των δεδομένων, που υποδεικνύει άμεσα τη θέση αποθήκευσης - όπως οι τίτλοι βιβλίων σε αυτήν την εργασία. Οι καλές συναρτήσεις κατακερματισμού διασφαλίζουν ότι η ίδια τιμή θα προκύπτει σε όσο το δυνατόν λιγότερες περιπτώσεις. Εάν συμβεί μια τέτοια σύγκρουση, η επιστήμη των υπολογιστών γνωρίζει καλές μεθόδους για να την αντιμετωπίσει.

Λέξεις κλειδιά και Websites

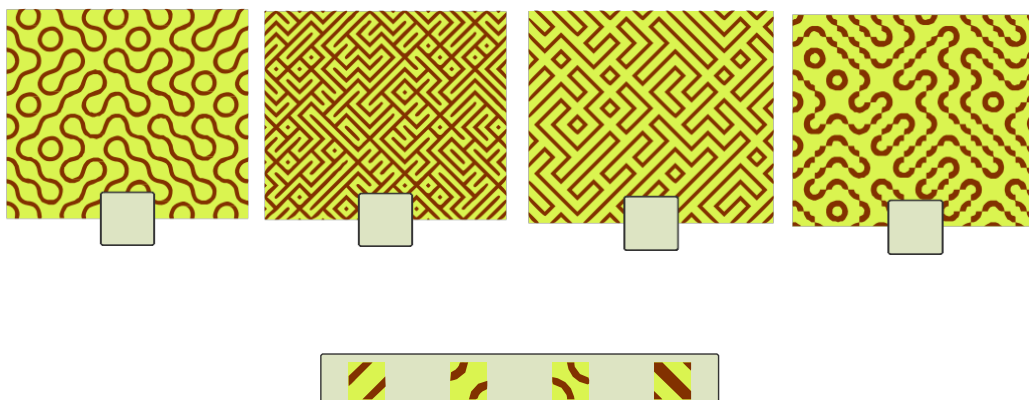
- Συνάρτηση κατακερματισμού (hash-function): https://en.wikipedia.org/wiki/Hash_function
Hashfunktion: <https://de.wikipedia.org/wiki/Hashfunktion>
- Πίνακας κατακερματισμού (hash table, hash map):
https://en.wikipedia.org/wiki/Hash_table
Hashtabelle: <https://de.wikipedia.org/wiki/Hashtabelle>
- http://www.abenteuer-informatik.de/PDF/ai2020_oa leseversion.pdf, Kapitel 11:
Ordnung im Chaos



19. Σχέδια από πλακίδια

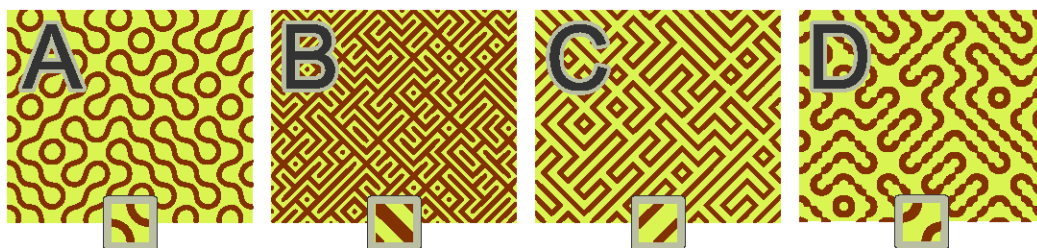
Τα ακόλουθα σχέδια δημιουργήθηκαν από ένα μόνο πλακίδιο, διαφορετικό όμως την κάθε φορά. Τα μεμονωμένα πλακίδια εμφανίζονται σε μεγέθυνση.

Αντιστοίχισε σχέδια και πλακίδια.

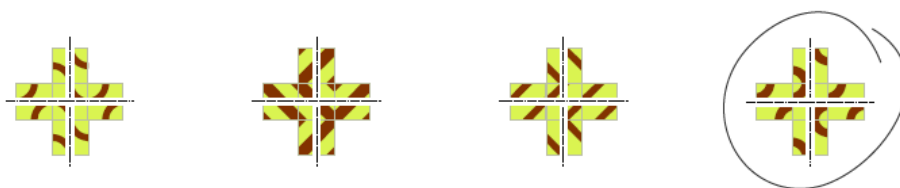


Λύση

Αυτή είναι η σωστή αντιστοίχιση:



Εάν βάλετε 5 πλακίδια το ένα δίπλα στο άλλο και τα συγκρίνετε πιο προσεκτικά, μπορείτε να δείτε σαφείς διαφορές:



Το πλακίδιο  είναι το μόνο πλακίδιο που έχει τέσσερις πλευρές που δεν ταιριάζουν ακριβώς μεταξύ τους. Αυτός είναι ο μόνος τρόπος για να δημιουργήσετε γραμμές διαφορετικού πλάτους, όπως στο σχέδιο D.

Το πλακίδιο  είναι το μόνο που μπορεί να δημιουργήσει τετραγωνισμένες κουκίδες όπως στο σχέδιο B, με τέσσερα εφάπτομενα τρίγωνα κάθε φορά. Επιπλέον, έχει το μεγαλύτερο ποσοστό εμβαδού καφέ σε σύγκριση με το κίτρινο, ακριβώς όπως φαίνεται στο πλακάκι B, αυτό δείχνει επίσης τη σχέση του συγκεκριμένου πλακιδίου με το σχέδιο B.

Το σχέδιο A φαίνεται να δημιουργείται από τις στρογγυλεμένες φόρμες του πλακιδίου  και οι ευθείες γραμμές του πλακιδίου  φαίνεται να παράγουν το σχέδιο C.

Αυτό είναι Πληροφορική!

Αυτά τα πλακίδια πήραν το όνομά τους από τον Sebastien Truchet (1657-1729) που εργάστηκε πάνω σε διάφορες παραλλαγές αυτών των πλακιδίων. Τα πλακίδια με 4 ίσες πλευρές σχηματίζουν ένα υποσύνολο πλακιδίων Truchet (αλλά τα πλακίδια Truchet δεν έχουν απαραίτητα 4 ίσες πλευρές, όπως σε 3 από τα παραπάνω σχέδια). Το γεγονός ότι πολύπλοκα μοτίβα μπορούν να δημιουργηθούν με πολύ απλά δομικά στοιχεία είναι μια ενδιαφέρουσα ιδιότητα που συναντάμε ξανά και ξανά στην επιστήμη των υπολογιστών. Τα πλακίδια Truchet μελετώνται στα μαθηματικά και την επιστήμη των υπολογιστών και χρησιμοποιούνται σε παιχνίδια υπολογιστών για τη δημιουργία λαβυρίνθων ή για τη δημιουργία διακοσμητικών μοτίβων.

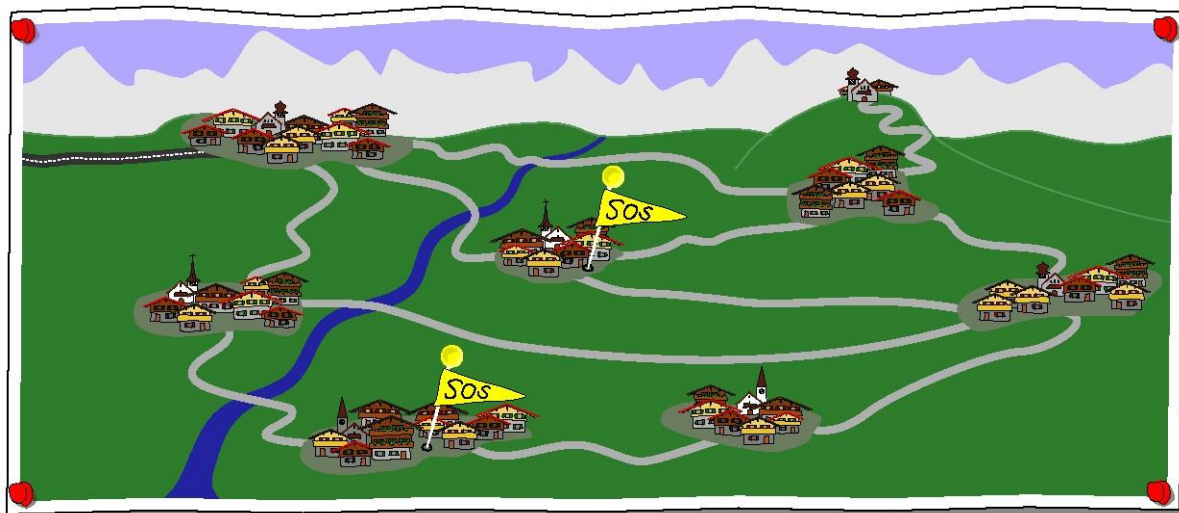
Λέξεις κλειδιά και Websites

Truchet tiles: https://en.wikipedia.org/wiki/Truchet_tiles



20. SOS από τα βουνά

Μερικά ορεινά χωριά τροφοδοτούνται από τη μεγάλη πόλη μέσω του παρακάτω οδικού δικτύου:



Μετά από μια καταιγίδα, αρκετά χωριά αναφέρουν ότι δεν είναι πλέον δυνατή η πρόσβαση σε αυτά, συγκεκριμένα εκείνα με τα σήματα SOS. Μπορούμε να συμπεράνουμε ότι κάποιοι δρόμοι είναι αποκλεισμένοι.

Για κάθε δρόμο μεταξύ χωριών σε αυτό το οδικό δίκτυο, να αναφέρετε εάν

(1) είναι αποκλεισμένος 

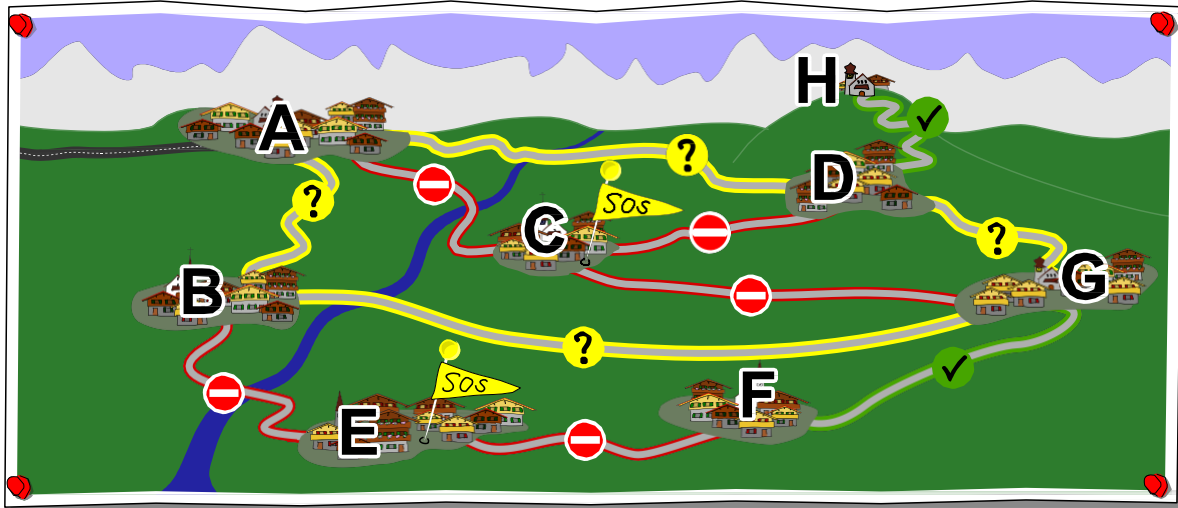
(2) είναι προσπελάσιμος  ή

(3) αν μπορούμε να πούμε χωρίς περαιτέρω πληροφορίες εάν ο δρόμος είναι προσπελάσιμος ή αποκλεισμένος 



Λύση

Ο χάρτης δείχνει τι γνωρίζουμε για τις συνδέσεις στο οδικό δίκτυο:



Ξεκινάμε με τον εντοπισμό των αποκλεισμένων δρόμων. Οι δύο δρόμοι που οδηγούν στο χωριό E ενδέχεται να είναι αποκλεισμένοι, διαφορετικά το χωριό E θα εξακολουθούσε να είναι προσβάσιμο. Ομοίως, οι τρεις δρόμοι προς το χωριό C είναι αποκλεισμένοι, διαφορετικά το χωριό C θα εξακολουθούσε να είναι προσβάσιμο.

Στη συνέχεια αναζητούμε τους δρόμους που πρέπει να είναι βατοί. Ο δρόμος μεταξύ των χωριών G και F πρέπει να είναι βατός, διαφορετικά, λόγω του αποκλεισμένου δρόμου μεταξύ των χωριών F και E, το χωριό F δεν θα ήταν προσβάσιμο. Ο δρόμος μεταξύ της εκκλησίας H και του χωριού D πρέπει επίσης να είναι βατός, καθώς το H είναι προσβάσιμο και μπορεί να φτάσει κανείς μόνο μέσω D.

Τώρα παραμένει να εξετάσουμε τους δρόμους που ίσως είναι βατοί. Δεδομένου ότι τα χωριά B, C και D συνδέονται με το χωριό A με πολλούς τρόπους, δεν μπορούμε να πούμε ποιος από τους υπόλοιπους δρόμους είναι βατός. Για παράδειγμα, στο χωριό B θα μπορούσαμε να πάμε μέσω του χωριού A, αλλά και μέσω του χωριού G. Το ίδιο ισχύει και για το χωριό D. Το χωριό G μπορεί να εφοδιαστεί είτε από το χωριό B είτε από το D. Οποιοσδήποτε από τους δρόμους στη διαδρομή A - B - G - D - A θα μπορούσε να είναι αποκλεισμένος και παραταύτα και τα 4 αυτά χωριά θα μπορούσαν να παραμείνουν όλα προσβάσιμα.



Αυτό είναι Πληροφορική!

Όπως και στα οδικά δίκτυα, έτσι και οι συνδέσεις στα δίκτυα υπολογιστών μπορεί επίσης να είναι ελαττωματικές, υπερφορτωμένες ή εντελώς κατεστραμμένες. Προκειμένου να αποφευχθούν αστοχίες, λαμβάνονται προληπτικά μέτρα ασφαλείας όπως π.χ. περισσότερες συνδέσεις για μια συγκεκριμένη τοποθεσία. Αυτό ονομάζεται *πλεονασμός δικτύου (network redundancy)*.

Ο εντοπισμός σφαλμάτων ενός συστήματος είναι μια εργασία που οι επιστήμονες υπολογιστών πρέπει να κάνουν πολύ συχνά, όχι μόνο στα δίκτυα υπολογιστών αλλά και στην ανάπτυξη λογισμικού. Για να διορθώσετε ένα σφάλμα, πρέπει να προσδιορίσετε την ακριβή πηγή του και αυτή η διαδικασία υλοποιείται ως επί το πλείστον σταδιακά και σε πολλά βήματα. Μερικοί προγραμματιστές πιστεύουν ότι ποτέ δεν μπορούν να βρεθούν όλα τα σφάλματα και τα bugs (απροσδόκητα προβλήματα στο software hardware ενός υπολογιστή) ενός προγράμματος.

Λέξεις κλειδιά και Websites

- Redundanz: [https://de.wikipedia.org/wiki/Redundanz_\(Technik\)](https://de.wikipedia.org/wiki/Redundanz_(Technik))
- Debuggen: <https://de.wikipedia.org/wiki/Debuggen>

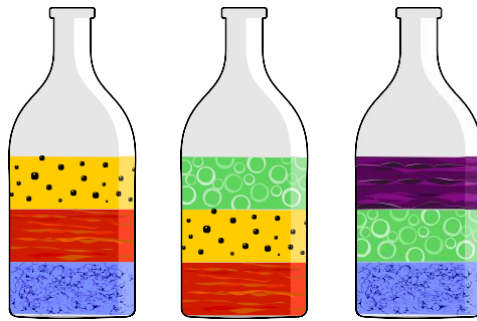




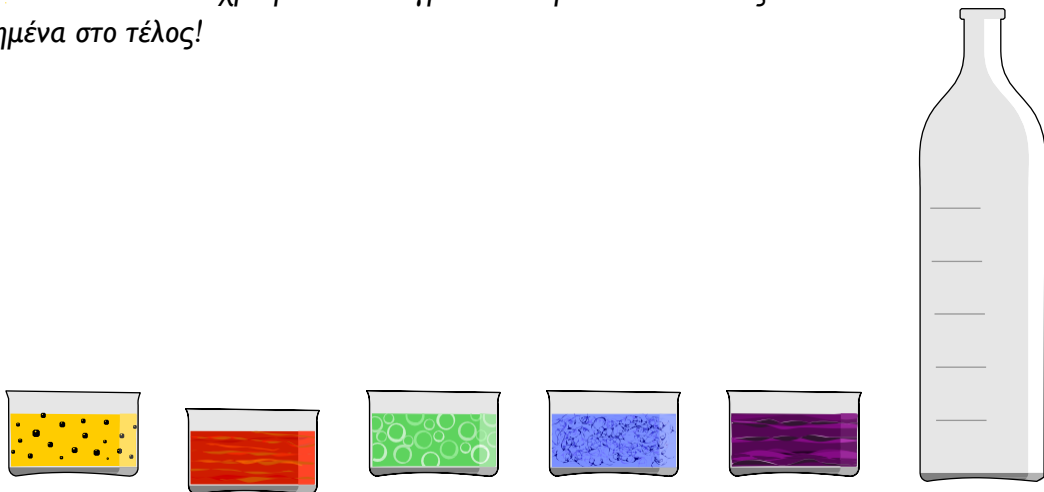


21. Στρώματα κατά πυκνότητα!

Ο Μάρκος έχει μπουκάλια με τρία χρωματιστά υγρά το καθένα, το κάθε υγρό βρίσκεται πάνω σε ένα άλλο. Γνωρίζει ότι τα υγρά με μικρότερη πυκνότητα κινούνται πάντα πάνω από υγρά με μεγαλύτερη πυκνότητα. Τώρα θέλει να δει πώς είναι όταν θα βάλει όλα τα χρωματιστά υγρά σε ένα μπουκάλι.



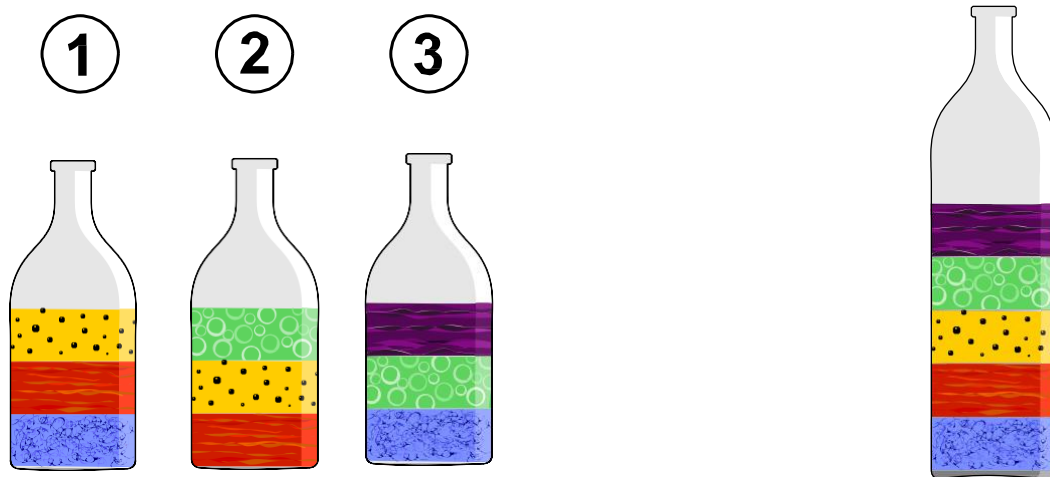
Τοποθετήστε τα πέντε χρωματιστά υγρά στο μπουκάλι όπως θα είναι τακτοποιημένα στο τέλος!





Λύση

Η εικόνα δείχνει τη σωστή διάταξη των πέντε χρωματιστών υγρών στο μεγάλο μπουκάλι.



Για να βρούμε τη σειρά των υγρών ακολουθούμε τα εξής: Βήμα-βήμα αφαιρούμε νοερά τα υγρά από τα τρία μπουκάλια που δίνονται και που δεν είναι πάνω από άλλο υγρό και τα βάζουμε στο μεγάλο μπουκάλι.

Στην αρχή μόνο τα δύο μπουκάλια 1 και 3 έχουν μπλε υγρό και εκεί είναι στο κάτω μέρος, επομένως δεν είναι πάνω από κανένα άλλο στρώμα υγρού. Το κόκκινο υγρό βρίσκεται στο μπουκάλι 2 στο κάτω μέρος. Αλλά στο μπουκάλι 1 είναι πάνω από το μπλε υγρό και επομένως πρέπει να έχει μικρότερη πυκνότητα από το μπλε. Έτσι το πρώτο πράγμα που πρέπει να κάνουμε, νοερά, είναι να αφαιρέσουμε το μπλε υγρό από τα μπουκάλια και να το βάλουμε στο μεγάλο μπουκάλι.

Τώρα το κόκκινο υγρό είναι το μόνο που δεν είναι πάνω από άλλο υγρό. Αφαιρείται από τα μπουκάλια 1 και 2 και τοποθετείται στο μεγάλο μπουκάλι. Μετά έρχεται το κίτρινο, μετά το πράσινο και τέλος το βιολετί υγρό, που έχει τη μικρότερη πυκνότητα και δεν καλύπτεται από κανένα άλλο.

Αυτό είναι Πληροφορική!

Στην επίλυση αυτής της εργασίας εκτιμήσαμε τη διάταξη των υγρών στα τρία μπουκάλια και ταξινομήσαμε τα υγρά ανάλογα με την πυκνότητά τους.

Μια ουσία έχει πολλές μετρήσιμες ιδιότητες: σημείο βρασμού, σημείο τήξης, ηλεκτρική αγωγιμότητα και πυκνότητα. Σε αυτή την περίπτωση, η πυκνότητα χρησιμοποιήθηκε ως κριτήριο για την ταξινόμηση των υλικών.



Η ταξινόμηση δεδομένων παίζει σημαντικό ρόλο σε πολλά προγράμματα υπολογιστών. Η μέθοδος που χρησιμοποιείται σε αυτήν την εργασία για τον προσδιορισμό της σειράς των υγρών στρωμάτων ονομάζεται *τοπολογική ταξινόμηση (topological sort)*. Χρησιμοποιείται για την ταξινόμηση αντικειμένων που ξέρουμε ότι έχουν σχέσεις του τύπου προκατόχου/διαδόχου.

Λέξεις κλειδιά και Websites

- Sortieren, Ordnung: <https://de.wikipedia.org/wiki/Sortierung>
- Topologisches Sortieren: https://de.wikipedia.org/wiki/Topologische_Sortierung, https://www.ac.tuwien.ac.at/files/teaching/ss12/AD1/top_sortieren.pdf





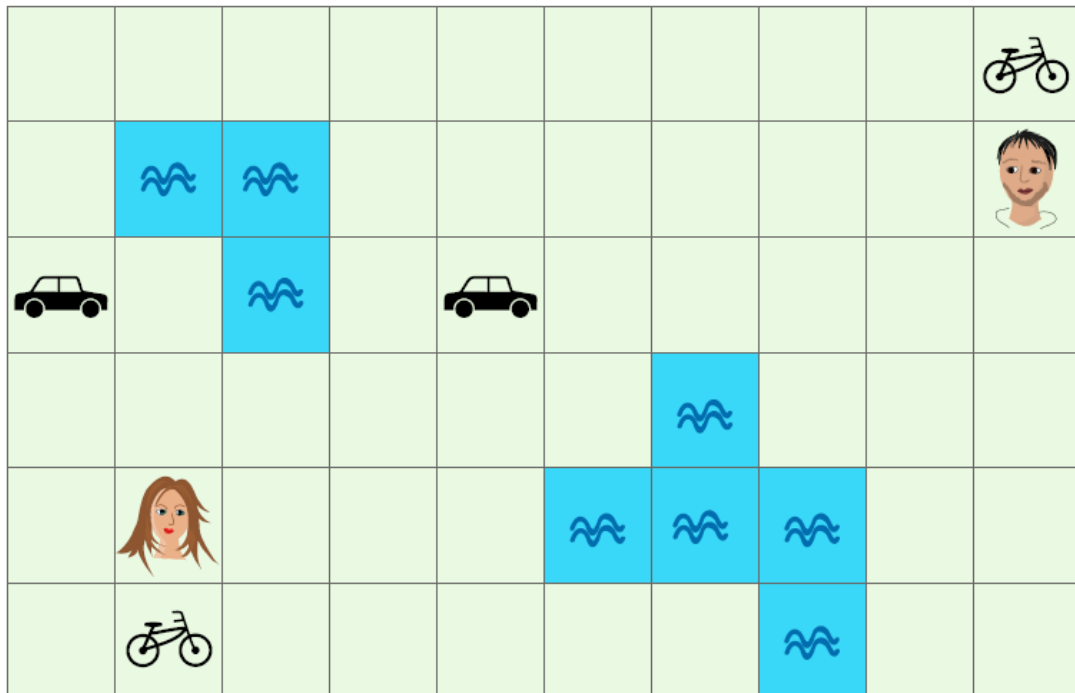


22. Είναι επείγον!

Δύο φίλοι θέλουν να συναντηθούν το συντομότερο δυνατόν. Από ένα τετράγωνο μπορούν να μετακινηθούν σε ένα διπλανό, αριστερά, δεξιά, πάνω ή κάτω.

Με τα πόδια η μετάβαση από το ένα τετράγωνο στο άλλο διαρκεί 1 λεπτό. Όταν φτάσουν σε έναν χώρο με όχημα, μπορούν να το χρησιμοποιήσουν.

Με ένα ποδήλατο μπορούν να καλύψουν 2 τετράγωνα σε ένα λεπτό και με ένα αυτοκίνητο 5 τετράγωνα σε ένα λεπτό. Αλλαγές κατεύθυνσης επιτρέπονται. Δεν μπορούν να διασχίσουν τα τετράγωνα με το νερό.



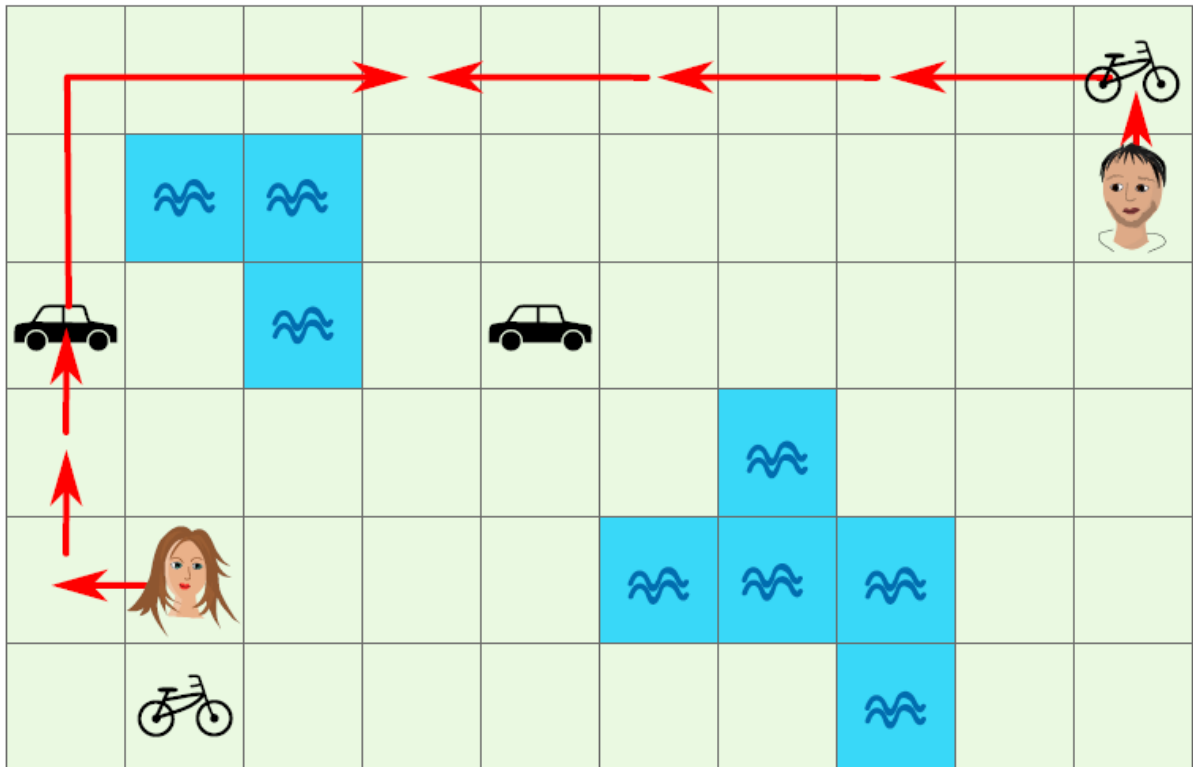
Πόσα λεπτά το λιγότερο χρειάζονται οι δύο φίλοι για να συναντηθούν σε ένα τετράγωνο?

- A) 1 λεπτό
- B) 2 λεπτά
- C) 3 λεπτά
- D) 4 λεπτά
- E) 5 λεπτά
- F) 6 λεπτά



Λύση

Η σωστή απάντηση είναι η D) 4 λεπτά. Minuten. Η εικόνα δείχνει μια διαδρομή έτσι ώστε οι δύο φίλοι να συναντηθούν σε 4 λεπτά.



Τώρα μένει να αποδειχθεί ότι δεν μπορούν να συναντηθούν σε 3 λεπτά: οι δύο φίλοι απέχουν 11 τετράγωνα. Σε 3 λεπτά όμως μπορούν να περπατήσουν μόνο 6 συνολικά τετράγωνα πιο κοντά το ένα στο άλλο. Αν ο ένας έχει φτάσει στο ποδήλατο και ο άλλος περπατάει, τότε σε 3 λεπτά μπορούν να πλησιάσουν 9 τετράγωνα το ένα στο άλλο, κάτι που ούτε αυτό είναι αρκετό. Ακόμα κι αν και οι δύο πάνε σε ποδήλατο, δεν είναι αρκετό. Γιατί τότε θα μπορούσαν να πλησιάσουν 12 τετράγωνα το ένα στο άλλο σε 3 λεπτά, αλλά τα δύο ποδήλατα απέχουν 13 τετράγωνα μεταξύ τους.

Άρα η μόνη επιλογή που απομένει είναι η χρήση ενός αυτοκινήτου. Σε 3 λεπτά μόνο η κοπέλα μπορεί να πάρει αυτοκίνητο. Αλλά τότε δεν υπάρχει άλλος χρόνος για να χρησιμοποιήσετε το αυτοκίνητο. Και σε 3 λεπτά το αγόρι δεν μπορεί να φτάσει σε τετράγωνο με αυτοκίνητο.

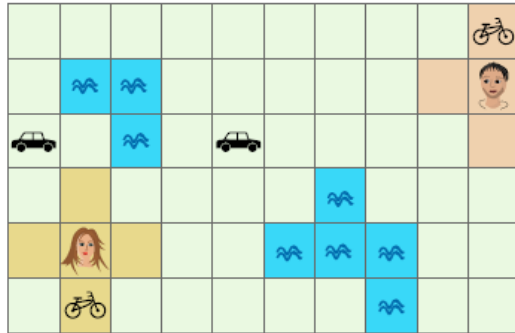
Αυτό είναι Πληροφορική!

Πώς λύσατε την άσκηση; Βρήκατε τυχαία μια γρήγορη διαδρομή και ελπίζατε να μην υπήρχε πιο γρήγορη; Ή δοκιμάσατε πολλές δυνατότητες και επισημάνατε την πιο γρήγορη;

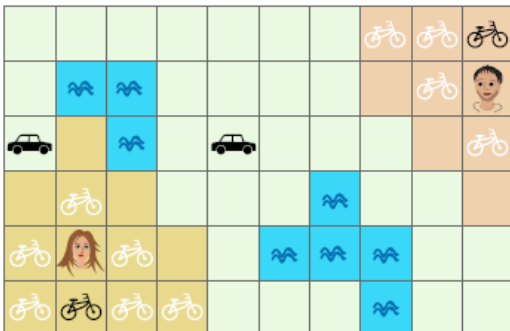




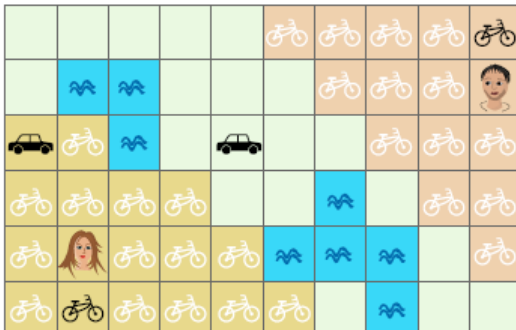
Τα προγράμματα υπολογιστών που έχουν αναπτυχθεί για αυτό το είδος προβλημάτων συνήθως λειτουργούν σύμφωνα με μια μέθοδο που ονομάζεται αναζήτηση κατά πλάτος (*breadth-first search, BFS*). Σε αυτήν την εργασία, η αναζήτηση κατά πλάτος λειτουργεί ως εξής:



1. Σημειώστε όλα τα τετράγωνα που μπορούν να φτάσουν οι δύο φίλοι σε ένα λεπτό.

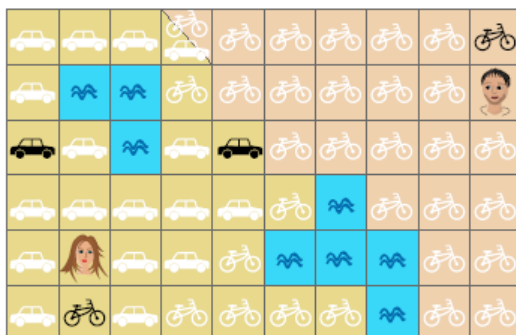


2. Σημειώστε όλα τα τετράγωνα που μπορούν να βρισκονται οι δύο φίλοι το πολύ σε ένα λεπτό ξεκινώντας από τα τετράγωνα που σημειώσατε στο βήμα 1. Σημειώστε επίσης τα μεταφορικά μέσα που χρησιμοποιούνται



3. Σημειώστε όλα τα τετράγωνα που μπορούν να βρισκονται οι δύο φίλοι, ξεκινώντας από τα τετράγωνα που σημειώσατε στο βήμα 2

Επειδή οι δύο περιοχές που έχετε σημειώσει μέχρι τώρα δεν αλληλοκαλύπτονται, οι φίλοι δεν μπορούν να συναντηθούν μετά από τρία λεπτά.



4. Τώρα σημειώστε όλα τα τετράγωνα στα οποία μπορείτε να φτάσετε σε ένα λεπτό ξεκινώντας από τα τετράγωνα που σημειώσατε στο βήμα 3





Τώρα οι δύο περιοχές αλληλεπικαλύπτονται σε ένα τετράγωνο. Είναι προσβάσιμο μετά από 4 λεπτά από το κορίτσι με αυτοκίνητο και από το αγόρι με ποδήλατο. Τα συστήματα πλοήγησης βρίσκουν την ταχύτερη διαδρομή μεταξύ δύο σημείων. Φροντίζουν ώστε η διαδρομή να ακολουθεί κατάλληλους δρόμους και μονοπάτια – και όχι «κόβοντας δρόμο» ή μέσα από ποτάμια. Αυτή η εργασία είναι παρόμοια με το πρόβλημα πλοήγησης, με τη διαφορά ότι εδώ δύο άτομα πρέπει να οδηγηθούν σε έναν κοινό -αρχικά άγνωστο- προορισμό και όχι μόνο ένα άτομο σε έναν σταθερό προορισμό.

Επειδή ένας υπολογιστής προχωρά συστηματικά σε αναζήτηση πλάτους, βρίσκει επίσης λύσεις που δεν είναι άμεσα εμφανείς. Μερικές φορές μια παράκαμψη με λιγότερα φανάρια είναι ταχύτερη από τη συντομότερη διαδρομή μεταξύ της αρχής και του προορισμού. Μια σύνδεση τρένου με μετεπιβιβάσεις μπορεί να είναι ταχύτερη από μια απευθείας σύνδεση με λεωφορείο. Στην επιστήμη των υπολογιστών, υπάρχουν διάφορες μέθοδοι για να βρεθεί η καλύτερη λύση σε ένα πρόβλημα αυτού του τύπου. Πέρα από την αναζήτηση κατά πλάτος, που μόλις περιγράφηκε, υπάρχει επίσης μια μέθοδος που ονομάζεται μέθοδος διακλάδωσης και οριοθέτησης (*Branch and Bound method*, *BB*, *B&B*, or *BnB method*).

Η αναζήτηση κατά πλάτος λαμβάνει υπόψη κάθε μερική λύση που μπορεί να επιτευχθεί με έναν ορισμένο αριθμό βημάτων εργασίας. Με το *Branch and Bound*, κανείς δεν επιδιώκει μερικές λύσεις εάν γνωρίζει ότι δεν μπορούν να οδηγήσουν στη βέλτιστη λύση.

Όταν ένα πρόβλημα γίνεται πολύ περίπλοκο, θα χρειαζόταν πολύς καιρός ακόμη και ο ταχύτερος υπολογιστής στον κόσμο για να τρέξει όλες τις δυνατότητες για να βρει την καλύτερη λύση. Στην πράξη, με ένα σύστημα πλοήγησης αρκεί συχνά να βρεθεί μια πολύ καλή διαδρομή, ακόμα κι αν δεν είναι η καλύτερη δυνατή. (Εάν μπορείτε να φτάσετε τον στόχο σας σε 78 λεπτά, πιθανότατα δεν σας πειράζει αν μπορούσατε θεωρητικά να τον φτάσετε σε 77 λεπτά.)

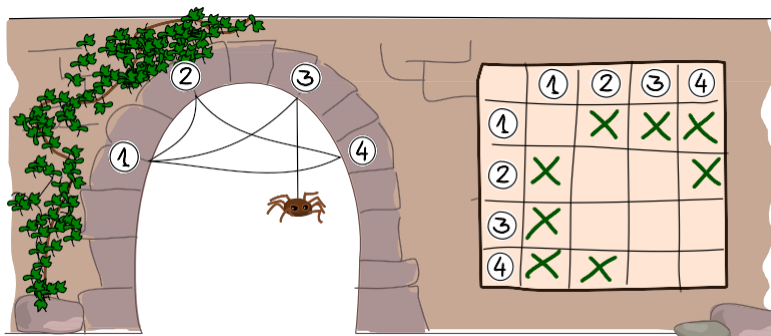
Λέξεις κλειδιά και Websites

- Breitensuche: <https://de.wikipedia.org/wiki/Breitensuche>
- Branch and Bound Algorithm: <https://de.wikipedia.org/wiki/Branch-and-Bound>



23. Τα δίχτυα της Θέκλας

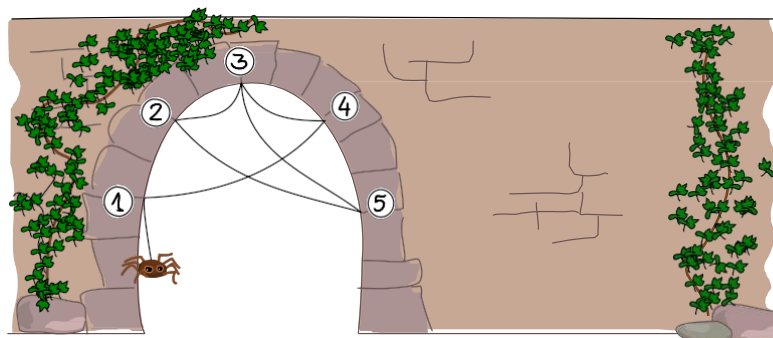
Η αράχνη Θέκλα θέλει να δημιουργήσει όσο το δυνατόν περισσότερους διαφορετικούς ιστούς. Γι' αυτό σκέφτηκε μια διαδικασία για να καταγράψει την ακριβή δομή των ιστών που φτιάχνει.



Να πώς το κάνει: Αριθμεί τα τελικά σημεία του ιστού από 1 έως N και χρησιμοποιεί κελιά σε έναν πίνακα σύμφωνα με τον ακόλουθο κανόνα:

- Εάν υπάρχει ένα νήμα που συνδέει το τελικό σημείο x με το τελικό σημείο y, τότε στο κελί του πίνακα γραμμή x, στήλη y σημειώνεται ένα X

Ένα νήμα που συνδέει το τελικό σημείο x με το τελικό σημείο y συνδέει επίσης το τελικό σημείο y με το τελικό σημείο x. Η Θέκλα χτίζει τώρα αυτόν τον ιστό:



Πώς καταγράφει η Θέκλα την κατασκευή αυτού του ιστού;

A)

	①	②	③	④	⑤
①				X	
②			X		X
③		X		X	X
④	X		X		
⑤		X	X		

B)

	①	②	③	④	⑤
①		X		X	
②	X		X		
③		X		X	X
④	X		X		
⑤			X		

C)

	①	②	③	④	⑤
①		X		X	
②			X		X
③		X		X	X
④	X		X	X	
⑤		X	X		

D)

	①	②	③	④	⑤
①				X	
②			X		X
③		X		X	X
④	X		X		
⑤			X		



Λύση

Η απάντηση Α είναι η σωστή γιατί όλα τα κελιά είναι σωστά σημειωμένα σύμφωνα με τον κανόνα

⑤		X	X		
---	--	---	---	--	--

Για την απάντηση Β, μια πρόσθετη σύνδεση σχεδιάστηκε λανθασμένα (τελικό σημείο 1 έως τελικό σημείο 2 και στις δύο κατευθύνσεις) και μία ξεχάστηκε (τελικό σημείο 2 έως τελικό σημείο 5 και στις δύο κατευθύνσεις).

	①	②	③	④	⑤
①		~~X~~		X	
②	~~X~~		X		X
③		X		X	X
④	X		X		
⑤		X	X		

Απάντηση C: Σύμφωνα με τον κανόνα που ορίζεται εδώ, δεν μπορεί να υπάρχουν σημάνσεις X στη διαγώνιο από πάνω αριστερά προς τα κάτω δεξιά. Εξάλλου, αυτό θα ήταν συνδέσεις μεταξύ ενός τελικού σημείου και του εαυτού του. Αν και αυτό θα μπορούσε κάλλιστα να επιτρέπεται σε ορισμένα δίκτυα, δεν συμβαίνει στον ιστό της αράχνης μας. Αλλά με την απάντηση C θα είχαμε 2 τέτοιες συνδέσεις (στο τελικό σημείο 1 και στο τελικό σημείο 4).

	①	②	③	④	⑤
①	~~X~~			X	
②			X		X
③		X		X	X
④	X		X	~~X~~	
⑤		X	X		

Απάντηση D: Όλες οι αναπαραστάσεις των δικτύων πρέπει να είναι συμμετρικές ως προς τη διαγώνιο από πάνω αριστερά προς τα κάτω δεξιά. Παρόλο που αυτή η απάντηση περιέχει τη σύνδεση από το τελικό σημείο 2 στο τελικό σημείο 5, η αντίστοιχη επιστροφή από το τελικό σημείο 5 στο τελικό σημείο 2 λείπει.

	①	②	③	④	⑤
①				X	
②			X		X
③		X		X	X
④	X		X		
⑤		X	X		

Αυτό είναι Πληροφορική!

Ο ιστός της αράχνης μπορεί να θεωρηθεί ως ένας γράφος (*graph*), μια έννοια που χρησιμοποιείται συχνά στην επιστήμη των υπολογιστών.

Ένας γράφος αποτελείται από *κόμβους* (*nodes*) (τα τελικά σημεία του δικτύου) και *ακμές* (*edges*) (τα νήματα μεταξύ των τελικών σημείων). Οι γράφοι χρησιμοποιούνται επίσης, για παράδειγμα, για την αναπαράσταση αντικειμένων και των σχέσεων μεταξύ αντικειμένων.

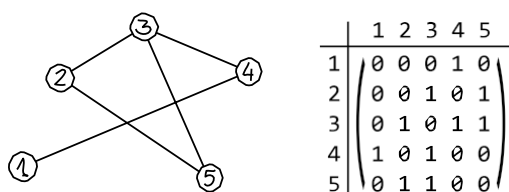




Για παράδειγμα, ένας γράφος θα μπορούσε να δείξει πώς οι άνθρωποι είναι φίλοι στα μέσα κοινωνικής δικτύωσης. Με ένα γράφο θα μπορούσαν να αναπαρασταθούν οι αεροπορικές πτήσεις μεταξύ χωρών.

Αυτή η άσκηση δείχνει πώς μπορείτε να αποθηκεύσετε τη δομή ενός ιστού αράχνης σε έναν πίνακα. Ορισμένες ιδιότητες, όπως η ακριβής γεωμετρική αναπαράσταση του δικτύου, χάνονται.

Σε πολλές περιπτώσεις όμως δεν ενδιαφέρεται κανείς για τις ακριβείς γεωμετρικές ιδιότητες ενός δικτύου, αλλά μόνο για τη δομή του. Οι βασικές πληροφορίες διατηρούνται: Πόσοι κόμβοι υπάρχουν; Και ανάμεσα σε ποια ζεύγη κόμβων υπάρχει μια ακμή;



Η δυνατότητα που παρουσιάζουμε εδώ είναι μόνο μία από τις πολλές δυνατότητες καταγραφής της δομής ενός δικτύου. Η μέθοδος δεν είναι πολύ «οικονομική» γιατί και οι δύο κατευθύνσεις αποθηκεύονται για κάθε σύνδεση, κάτι που δεν θα ήταν απαραίτητο. Επίσης ούτε τα διαγώνια κελιά είναι απαραίτητα. Από την άλλη πλευρά, αυτή η μέθοδος έχει το πλεονέκτημα ότι τα σφάλματα αναπαράστασης μπορούν εν μέρει να αναγνωριστούν. Για παράδειγμα, η απάντηση C και η απάντηση D θα μπορούσαν να αναγνωριστούν ως λανθασμένες χωρίς αναφορά στο δίκτυο.

Ο τρόπος αναπαράστασης που χρησιμοποιήθηκε εδώ καλείται *πίνακας γειτνίασης* (*adjacency matrix*).

Λέξεις κλειδιά και Websites

- πίνακας γειτνίασης (*adjacency matrix*): https://en.wikipedia.org/wiki/Adjacency_matrix
Adjazenzmatrix: <https://de.wikipedia.org/wiki/Adjazenzmatrix>





24. Φρούτο με φρούτο

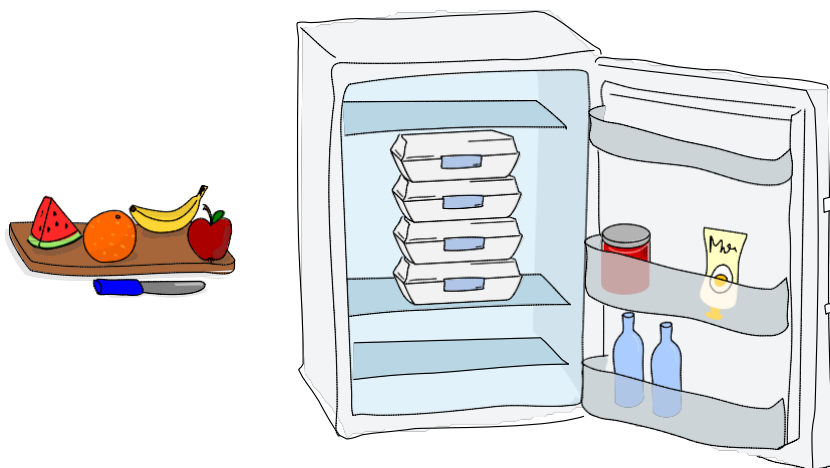
Ο μπαμπάς Κάστορας, η μαμά, η Ντόρι και ο Ρον Κάστορας ετοιμάζουν τέσσερα κουτιά πρωινού για την επόμενη μέρα, το καθένα με διαφορετικό φρούτο: μήλο, μπανάνα, πορτοκάλι και καρπούζι. Τα κουτιά στοιβάζονται το ένα πάνω στο άλλο στο ψυγείο. Το πρωί οι κάστορες είναι ακόμα πολύ κουρασμένοι και όταν φεύγουν από το λαγούμι παίρνουν απλώς το επάνω κουτί χωρίς να το κοιτάζουν πιο προσεκτικά.

Δεν είναι γνωστό ακριβώς με ποια σειρά οι κάστορες φεύγουν από το λαγούμι, αλλά σίγουρα η μαμά Κάστορα φεύγει πριν από την Ντόρι και ο μπαμπάς Κάστορας φεύγει πάντα τελευταίος.

Στα μέλη της οικογένειας αρέσουν διαφορετικά φρούτα. Ο πίνακας δείχνει τι αρέσει σε κάθε μέλος της οικογένειας.

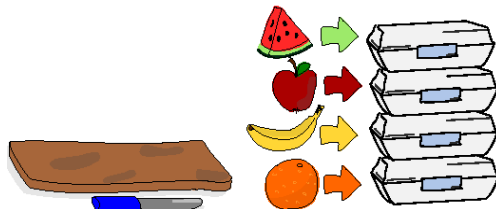
				
Μπαμπάς	—	—	✓	—
Μαμά	✓	—	✓	✓
Ντόρι	✓	✓	✓	—
Ρον	✓	✓	—	✓

Βάλτε τα φρούτα στα κουτιά έτσι ώστε όλοι οι κάστορες να πάρουν ένα κουτί με φρούτα που τους αρέσουν.



Λύση

Υπάρχει μόνο ένας τρόπος να μοιράσετε τα φρούτα με τέτοιο τρόπο ώστε ο καθένας σίγουρα να πάρει κάτι που του αρέσει:



Στον μπαμπά αρέσουν μόνο τα πορτοκάλια και είναι ο τελευταίος που φεύγει. Έτσι τα πορτοκάλια μπαίνουν στο κάτω κουτί. Ο Ρον πηγαίνει στο ψυγείο πρώτος, δεύτερος ή τρίτος. Η μαμά πηγαίνει επίσης πριν από την Ντόρι, οπότε η σειρά είναι ξεκάθαρη γνωρίζοντας τότε ο Ρον φεύγει από το λαγούμι. Οι ακόλουθες τρεις ακολουθίες είναι δυνατές:

- | | | |
|------------|---------|---------|
| 1. μαμά | μαμά | Ρον |
| 2. Ντόρι | Ρον | Μαμά |
| 3. Ρον | Ντόρι | Ντόρι |
| 4. μπαμπάς | μπαμπάς | μπαμπάς |

Έτσι, η Μαμά, η Ντόρι και ο Ρον θα μπορούσαν να φύγουν όλοι ως δεύτεροι. Το δεύτερο κουτί πρέπει επομένως να περιέχει ένα φρούτο που αρέσει και στους τρεις, και μόνο το μήλο το πληροί αυτό. Έτσι για το πάνω κουτί μένουν μόνο μπανάνες και καρπούζι. Δεδομένου ότι η μαμά δεν αγαπά τις μπανάνες, πρέπει να βάλουμε το καρπούζι. Έτσι οι μπανάνες μένουν για το τρίτο κουτί.

Αυτό είναι Πληροφορική!

Η σωστή σειρά παίζει πολύ μεγάλη σημασία σε πολλούς τομείς της επιστήμης των υπολογιστών: Πολλοί υπολογισμοί απαιτούν ενδιάμεσα αποτελέσματα που πρέπει πρώτα να υπολογιστούν πριν υπολογιστεί το τελικό αποτέλεσμα. Εάν τα βήματα υπολογισμού εκτελούνται σε διαφορετικούς υπολογιστές, θα μπορούσε χωρίς προσεκτικό σχεδιασμό να προκύψουν τα λεγόμενα *αδιέξοδα* (*deadlocks*). Αυτές είναι καταστάσεις όπου δύο ή περισσότεροι υπολογιστές περιμένουν ο ένας τον άλλον και έτσι το πρόγραμμα δεν τελειώνει ποτέ. Η λάθος σειρά οδηγεί συνήθως σε λάθη (όπως με τους κάστορες στη δυσαρέσκεια για το αγαπημένο φρούτο). Αν για παράδειγμα πρέπει να υπολογίσουμε τον τύπο $Z \leftarrow (A+B) \cdot (A-B)$, αυτό μπορεί να χωριστεί στα ακόλουθα βήματα ενός προγράμματος:





Εισαγωγή A

Εισαγωγή B

Υπολόγισε $X \leftarrow A + B$

Υπολόγισε $Y \leftarrow A - B$

Υπολόγισε $Z \leftarrow X \cdot Y$

Αλλά αν δοκιμάσετε για παράδειγμα, την εκτέλεση του βήματος υπολογισμού $Z \leftarrow X \cdot Y$ προτού υπολογίσετε το X, αυτό θα οδηγήσει σε σφάλμα και το πρόγραμμα θα διακοπεί. Ή θα χρησιμοποιηθεί μια προεπιλεγμένη τιμή για το X, η οποία συνήθως οδηγεί σε εσφαλμένο αποτέλεσμα. Επομένως, κατά τον προγραμματισμό, η σειρά με την οποία εκτελούνται οι εντολές είναι πολύ σημαντική.

Λέξεις κλειδιά και Websites

- αδιέξοδο (Deadlock): <https://en.wikipedia.org/wiki/Deadlock>
[https://de.wikipedia.org/wiki/Deadlock_\(Informatik\)](https://de.wikipedia.org/wiki/Deadlock_(Informatik))

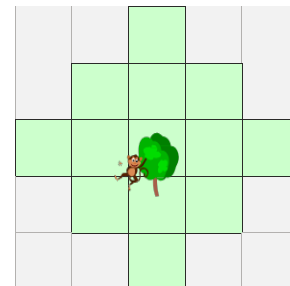




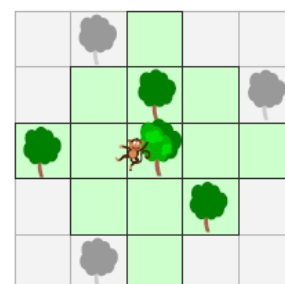


25. Η μαϊμουδίτσα Κοκο σκαρφαλώνει

Η μαϊμουδίτσα Κοκο που είναι πάνω στο δένδρο μπορεί να πηδήξει τόσο μακριά όσο ορίζει η πράσινη περιοχή.

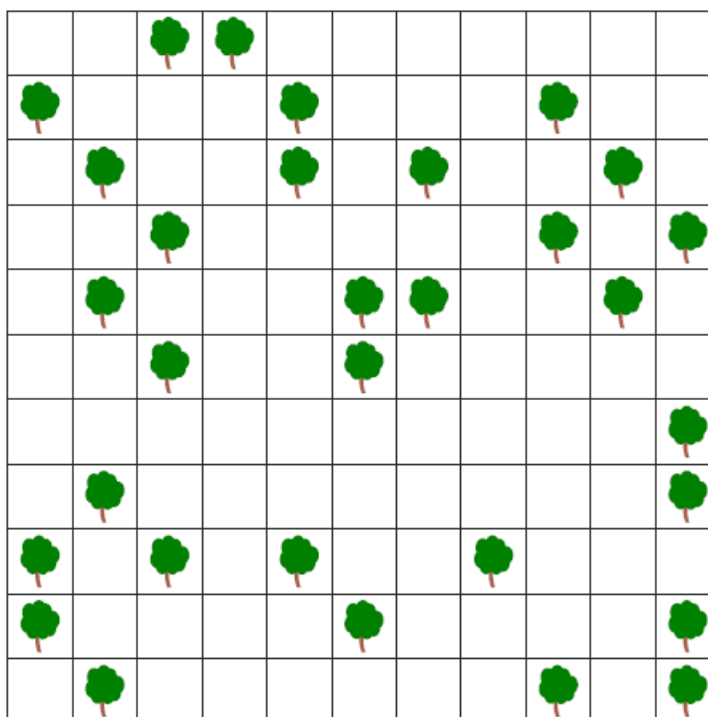


Στο παράδειγμα, η Κοκο φτάνει στα χρωματιστά δέντρα με ένα άλμα. Τα δύο γκρίζα δέντρα παραπάνω είναι προσβάσιμα επίσης με δύο άλματα, αλλά όχι το γκρίζο δέντρο από κάτω.



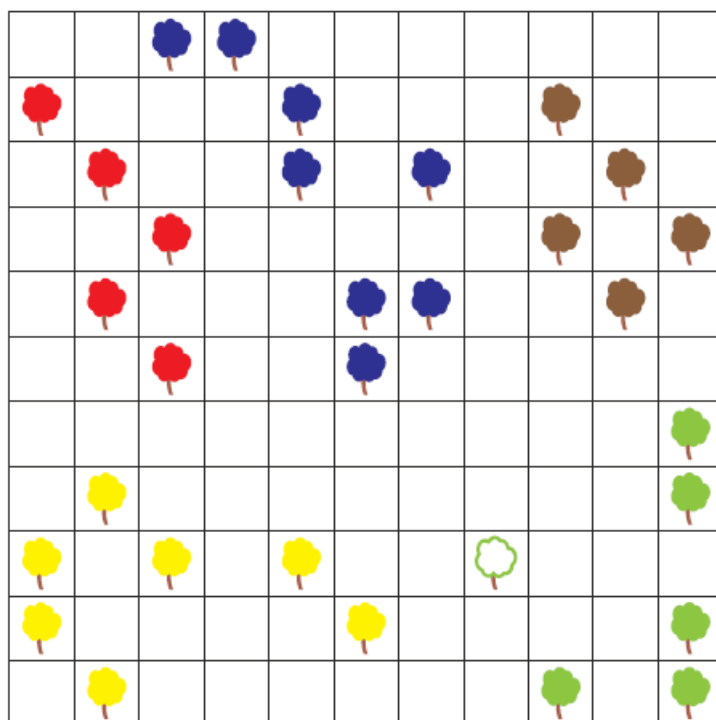
Υπάρχουν ομάδες από δέντρα, μεταξύ των οποίων η Κοκο μπορεί να μετακινείται με περισσότερα πηδήματα, χωρίς να χρειαστεί να ακουμπήσει στο έδαφος.

Σημειώστε όλα τα δέντρα της μεγαλύτερης τέτοιας ομάδας.



Λύση

Στην παρακάτω εικόνα, δύο δέντρα έχουν το ίδιο χρώμα εάν η Κoko μπορεί να φτάσει από το ένα στο άλλο χωρίς να αγγίζει το έδαφος.



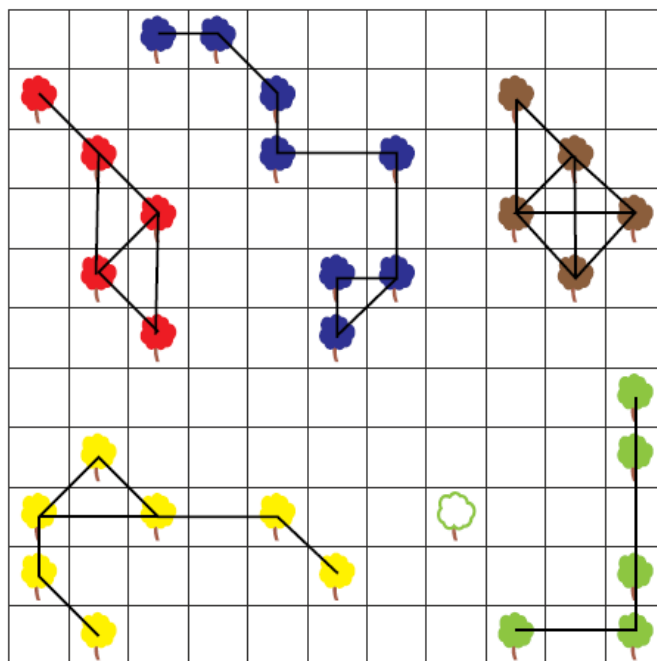
Όπως βλέπουμε η ομάδα με τα μπλε δένδρα είναι η μεγαλύτερη με 8 δέντρα.

Αυτό είναι Πληροφορική!

Εάν η Κoko μπορεί να πηδήξει από το ένα δέντρο απευθείας στο άλλο, τότε αυτά συνδέονται μεταξύ τους. Αυτό μπορούμε να το αναπαραστήσουμε ως γραμμή μεταξύ των δέντρων, όπως φαίνεται παρακάτω. Έτσι έχουμε ένα γράφο με δέντρα ως κόμβους και ακμές μεταξύ συνδεδεμένων δέντρων. Η Κoko μπορεί να μεταπηδήσει από το ένα δέντρο στο άλλο εάν και μόνο εάν οι ακμές σχηματίζουν μια διαδρομή μεταξύ των δύο δέντρων.

Ονομάζουμε μια ομάδα κόμβων *συνδεδεμένους (connected)* αν όλοι είναι συνδεδεμένοι με ακμές. Εάν δεν μπορούμε πλέον να κάνουμε μια τέτοια ομάδα μεγαλύτερη χωρίς να χάσουμε τη σύνδεση μεταξύ τους, τότε μιλάμε για ένα *συνδετικό στοιχείο ή συνεκτική συνιστώσα (connected component)*. Ένας γράφος μπορεί να χωριστεί σαφώς σε τέτοια συνδετικά στοιχεία, σημειώνονται με χρώμα στο παρακάτω σχήμα.





Ένα συνδετικό στοιχείο μπορεί να προσδιοριστεί απλά αρχίζοντας από οποιονδήποτε κόμβο και στη συνέχεια αναζητούνται οι κόμβοι που μπορούν να προσεγγιστούν μέσω των ακμών.

Λέξεις κλειδιά και Websites

- Συνεκτικότητα γράφου (Connectivity), συνεκτική συνιστώσα (connected component):
[https://en.wikipedia.org/wiki/Connectivity_\(graph_theory\)](https://en.wikipedia.org/wiki/Connectivity_(graph_theory))
- Zusammenhangskomponenten,
[https://de.wikipedia.org/wiki/Zusammenhang_\(Graphentheorie\)](https://de.wikipedia.org/wiki/Zusammenhang_(Graphentheorie))





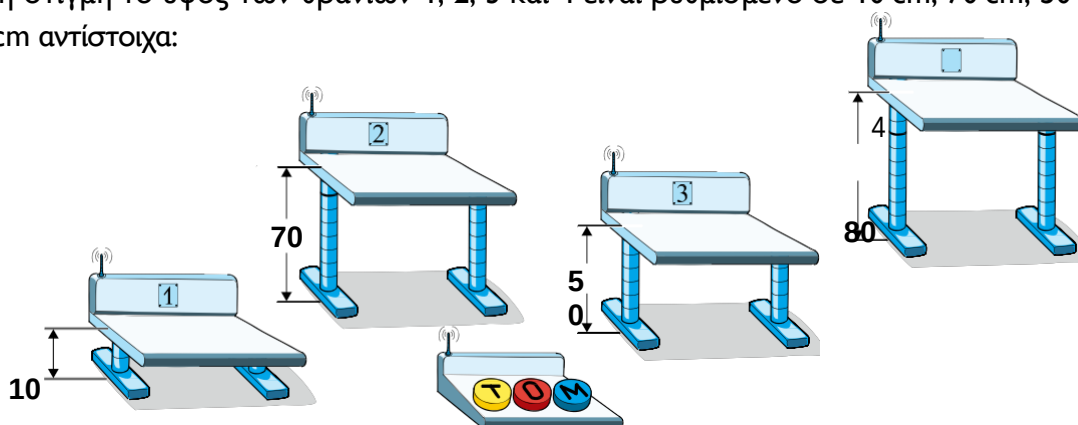
26. Τρελλο-θρανία

Στην τάξη υπάρχουν θρανία με ηλεκτρικά ρυθμιζόμενο ύψος. Όλα τα θρανία πρέπει να ρυθμιστούν σε ύψος 60 cm για το μάθημα. Το ύψος των ποδιών μπορεί να αλλάξει με τα κουμπιά ⚡, ⬇️, ⬆️ και ένα τηλεχειριστήριο. Κάποιος έπαιξε με το τηλεχειριστήριο και άλλαξε τις αρχικές ρυθμίσεις. Τώρα τα τρία κουμπιά λειτουργούν ως εξής:

- ⚡ υψώνει τα θρανία 1, 2 και 3 κάθε φορά 10 cm.
- ⬇️ βυθίζει τα θρανία 2, 3 und 4 κάθε φορά 10 cm.
- ⬆️ υψώνει τα θρανία 1, 3 und 4 κάθε φορά 10 cm.

Αυτές οι ενέργειες γίνονται κάθε φορά που θα πατηθεί το αντίστοιχο κουμπί.

Αυτή τη στιγμή το ύψος των θρανίων 1, 2, 3 και 4 είναι ρυθμισμένο σε 10 cm, 70 cm, 50 cm και 80 cm αντίστοιχα:



Πώς θα μπορούσε το ύψος όλων των θρανίων να ρυθμιστεί στα 60 cm?

- A) Πάτησε 4 x ⚡, 5 x ⬇️ και 1 x ⬆️
 B) Πάτησε 5 x ⚡, 1 x ⬇️ και 0 x ⬆️
 C) Πάτησε 3 x ⚡, 4 x ⬇️ και 2 x ⬆️
 D) Πάτησε 2 x ⚡, 4 x ⬇️ και 6 x ⬆️



Λύση

Η σωστή απάντηση είναι η C) πάτησε $3 \times$ , $4 \times$  και $2 \times$ . Στο τηλεχειριστήριο θα παρατηρήσετε ότι και τα τρία πλήκτρα αλλάζουν το ύψος κατά 10 cm κάθε φορά που τα πατάτε, δηλαδή πάντα με την ίδια τιμή. Δύο από τα πλήκτρα ( και ) υψώνουν το θρανίο ενώ ένα πλήκτρο το χαμηλώνει (). Επιπλέον, τα τρία πλήκτρα αλλάζουν το ύψος τριών θρανίων κάθε φορά, οπότε υπάρχει πάντα ένα θρανίο του οποίου το ύψος παραμένει αμετάβλητο. Το πλήκτρο  δεν επηρεάζει καθόλου το θρανίο 1, οπότε δεν μπορούμε να χαμηλώσουμε καθόλου το θρανίο 1.

Το θρανίο 1 είναι κατά 50 cm χαμηλωμένο. Από αυτό συμπεραίνουμε ότι πρέπει να πατήσουμε το πλήκτρο  ή το πλήκτρο  ακριβώς 5 φορές (ο συνολικός αριθμός των πατημάτων πλήκτρων στο  ή στο  πρέπει να είναι ακριβώς 5). Αυτό μπορεί να εκφραστεί ως εξίσωση: $T + M = 5$. Μπορούμε στη συνέχεια να αποκλείσουμε τη λύση D), επειδή εκεί ισχύει $T + M = 8$. Μετά το πάτημα της ακολουθίας πλήκτρων της λύσης D), το γραφείο 1 θα έχει ύψος $10 + 20 + 60 = 90$ cm, δηλαδή το αρχικό ύψος 10 cm συν 2×10 cm για το πλήκτρο  συν 6×10 cm για το πλήκτρο .

Το θρανίο 2 είναι κατά 10 cm πιο ψηλό. Το πλήκτρο  δεν επηρεάζει το θρανίο 2. Έτσι ισχύει για τη σωστή λύση $T - O = -1$. Με αυτό μπορούμε να αποκλείσουμε τη λύση B), γιατί τότε το θρανίο 2 θα είχε ύψος στο τέλος $70 + 50 - 10 = 110$.

Το θρανίο 3 είναι κατά 10cm χαμηλότερο, άρα ισχύει $T - O + M = 1$. Έτσι οι λύσεις A) και B) αποκλείονται. Στο A) το ύψος του γραφείου 3 θα ήταν και πάλι το ίδιο στο τέλος, δηλαδή $50 + 40 - 50 + 10 = 50$ cm, στο B) το ύψος του γραφείου 3 στο τέλος θα ήταν $50 + 50 - 10 = 90$ cm. Έως τώρα όλες οι λύσεις έχουν αποκλειστεί εκτός από τη λύση Γ).

Ωστόσο, πρέπει ακόμα να ελεγχθεί εάν η λύση C) οδηγεί επίσης στο σωστό ύψος για το θρανίο 4. Το θρανίο 4 είναι 20 cm πιο ψηλό και το πλήκτρο  δεν επηρεάζει το ύψος του γραφείου 4. Πρέπει λοιπόν να πατηθεί δύο φορές το πλήκτρο  και για κάθε πάτημά του  χρειάζεται ένα επιπλέον πάτημα . Με την ακολουθία πατήματος των πλήκτρων της λύσης C) το ύψος του γραφείου 4 στο τέλος είναι $80 - 40 + 20 = 60$ cm.

Αρχικά θεωρήσαμε ότι η λύση ότι η λύση C) λειτουργεί για τα θρανία 1, 2 και 3, τώρα είμαστε βέβαιοι ότι αυτή η λύση θα λειτουργήσει.

Εναλλακτικά, η λύση μπορεί να αναζητηθεί χρησιμοποιώντας τέσσερις γραμμικές εξισώσεις. Για κάθε γραφείο σημειώνετε με μια εξίσωση, ποια πλήκτρα αλλάζουν το ύψος του γραφείου και ποια είναι η αλλαγή ύψους που αναζητάτε. Για παράδειγμα, το ύψος του γραφείου 1 αλλάζει μόνο με  και , και η επιθυμητή ρύθμιση ύψους είναι 50 cm, η οποία μπορεί να επιτευχθεί με 5 πατήματα πλήκτρων (10 cm ανά πάτημα πλήκτρου).

Εφόσον υπάρχουν τέσσερα θρανία και τρία πλήκτρα, υπάρχουν τέσσερις γραμμικές εξισώσεις με τρεις αγνώστους:



$$T + M = 5$$

$$T - O = -1$$

$$T - O + M = 1$$

$$-O + M = -2$$

Εάν αφαιρέσουμε την τρίτη εξίσωση από την πρώτη παίρνουμε $O = 4$. Αντικαθιστούμε το O στη δεύτερη εξίσωση και παίρνουμε $T = 3$. Τώρα με $M = 2$ ισχύουν όλες οι εξισώσεις. Άρα είναι η μόνη λύση

Αυτό είναι Πληροφορική!

Αυτή είναι μια τυπική εργασία από το πεδίο της διακριτής βελτιστοποίησης (*discrete optimization*), πιο συγκεκριμένα του γραμμικού προγραμματισμού (*linear programming*). Σε αυτή την άσκηση δίνεται ένα σύνολο περιορισμών. Σε αυτή την ειδική περίπτωση, μπορούν όλοι να διατυπωθούν ως γραμμικές εξισώσεις. Ο στόχος είναι τυπική περίπτωση για την επιστήμη των υπολογιστών. Ψάχνουμε για μια σειρά ενεργειών που οδηγούν σε έναν προκαθορισμένο στόχο. Θα μπορούσε κανείς ακόμη να περιγράψει την όλη εργασία ως αναζήτηση μονοπατιού σε έναν τετραδιάστατο χώρο με τρεις επιτρεπόμενες ενέργειες, δηλαδή από το σημείο (10, 70, 50, 80) στο σημείο (60, 60, 60, 60). Σε αυτήν την εργασία υπάρχει μόνο μία λύση, αλλά τέτοιες εργασίες έχουν συχνά πολλές λύσεις, οπότε τίθεται ως στόχος η βελτιστοποίηση (*optimization*). Στη συνέχεια αναζητείται το ελάχιστο της γραμμικής συνάρτησης $T + M + O$.

Λέξεις κλειδιά και Websites

- διακριτή βελτιστοποίηση (*discrete optimization*), ακέραιος γραμμικός προγραμματισμός (*integer linear programming*):
https://en.wikipedia.org/wiki/Discrete_optimization
Diskrete Optimierung, ganzzahlige lineare Optimierung:
https://en.wikipedia.org/wiki/Integer_programming
https://de.wikipedia.org/wiki/Ganzzahlige_lineare_Optimierung
- Πρόβλημα Προσβασιμότητας (σε γράφους), (Undirected/directed reachability problem)
[https://en.wikipedia.org/wiki/SL_\(complexity\)](https://en.wikipedia.org/wiki/SL_(complexity))
Erreichbarkeit in gerichteten Konfigurationsgraphen:
https://de.wikipedia.org/wiki/Erreichbarkeitsproblem_in_Graphen





27. Η ταινία με τις μπίλιες

Ο κάστορας κινείται σε μια ταινία από αριστερά προς τα δεξιά, ανά ένα τετράγωνο κάθε φορά. Σε κάθε τετράγωνο της κορδέλας μπορεί να υπάρχει μία μπίλια.

Κάθε φορά που ο κάστορας πατάει σε ένα τετράγωνο και έχει τα χέρια του ελεύθερα, παίρνει τη μπίλια και μετά την κουβαλάει μαζί του.

Στο πρώτο ελεύθερο τετράγωνο αφήνει κάτω την μπίλια
Ο κάστορας μπορεί να κουβαλήσει μόνο μία μπίλια και σε κάθε τετράγωνο υπάρχει χώρος μόνο για μία μπίλια.

Όταν ο κάστορας κουβαλάει ήδη μία μπίλια και πατάει σε τετράγωνο που υπάρχει άλλη μπίλια...

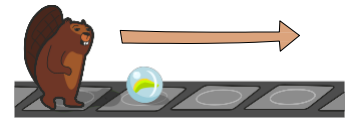
... τότε προσπερνάει το τετράγωνο.

... και αφήνει την μπίλια του στο επόμενο ελεύθερο τετράγωνο
Μετά μπορεί να σηκώσει την επόμενη μπίλια.

Ο κάστορας στέκεται μπροστά σε μια περιοχή της ταινίας που έχει πάνω της τρεις μπίλιες. Σε ποια τετράγωνα θα βρίσκονται οι μπίλιες όταν ο κάστορας θα έχει διασχίσει την ταινία;



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

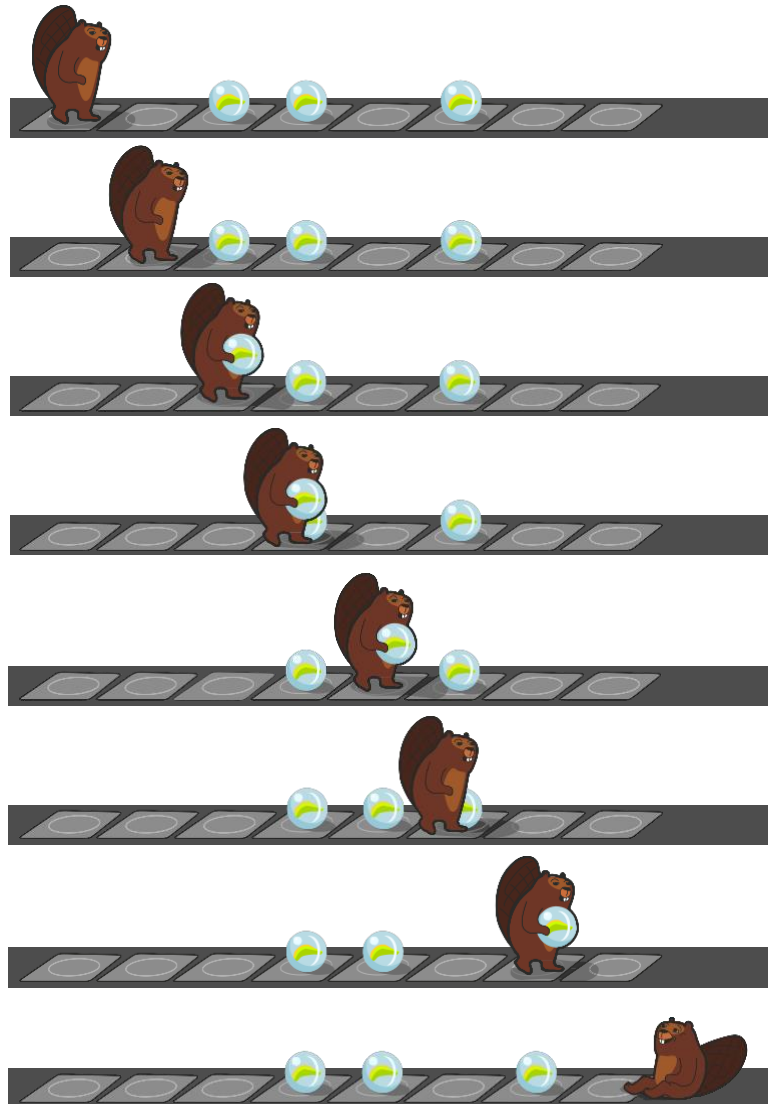


Λύση

Η σωστή απάντηση είναι η C.



Το ακόλουθο γραφικό δείχνει τις ενέργειες:



Αυτό είναι Πληροφορική!

Στην επιστήμη των υπολογιστών, σχετικά απλές λειτουργίες συχνά οδηγούν σε ενδιαφέροντα αποτελέσματα. Αυτή η άσκηση είναι ένα καλό παράδειγμα. Ο κάστορας ακολουθεί έναν αλγόριθμο (*algorithm*). Ο αλγόριθμος αυτός βασίζεται στο γεγονός ότι ο κάστορας μπορεί να έχει δύο διαφορετικές καταστάσεις (κουβαλάει μία μπίλια ή όχι) και



ότι μπορεί να βρει δύο διαφορετικούς τύπους τετραγώνων στο δρόμο του (κατειλημμένο και άδειο).

Όσο απλό κι αν είναι το παράδειγμα σε αυτό το πρόβλημα του κάστορα, περιέχει όμως μερικά από τα βασικά στοιχεία μιας μηχανής Turing. Μια μηχανή Turing (που πήρε το όνομά της από τον μαθηματικό Alan Turing) είναι ένας ειδικός, πολύ απλά δομημένος υπολογιστής. Βασικά μια μηχανή Turing μπορεί να εκτελέσει οποιονδήποτε αλγόριθμο που μπορεί να εκτελέσει ένας συμβατικός υπολογιστής. Στην πράξη, ωστόσο, οι μηχανές Turing δεν χρησιμοποιούνται ως υπολογιστές γιατί μπορούμε να κατασκευάσουμε υπολογιστές που είναι ναι μεν πιο περίπλοκοι αλλά πολύ πιο αποτελεσματικοί. Οι μηχανές Turing είναι κυρίως χρήσιμες στη θεωρία. Λόγω της απλής δομής τους, είναι σχετικά εύκολο να αποδειχθούν θεωρίες με τις μηχανές Turing. Και αν μπορεί να αποδειχθεί ότι ένα πρόβλημα δεν μπορεί να λυθεί από τις μηχανές Turing, τότε δεν μπορεί να λυθεί από κανέναν υπολογιστή.

Μια μηχανή Turing αποτελείται από:

- Μία ταινία (*tape*) μεταβλητού μήκους που χωρίζεται σε κελιά (*cells*), το ένα δίπλα στο άλλο. Κάθε κελί περιέχει ένα σύμβολο. Στο παράδειγμά μας, τα κελιά είναι τα τετράγωνα στα οποία κινείται ο κάστορας.
- Ένα πεπερασμένο σύνολο συμβόλων (*symbols*). Συχνά χρησιμοποιούνται ως σύμβολα μόνο το 0 και το 1. Στο παράδειγμά μας η ύπαρξη μπίλιας αντιπροσωπεύει το 1 και ένα κενό τετράγωνο αντιπροσωπεύει το 0.
- Μια κεφαλή (*head*) ανάγνωσης/εγγραφής που κινείται και προς τις δύο κατευθύνσεις στην ταινία και, με αυτόν τον τρόπο, μπορεί να διαβάσει σύμβολα στην ταινία και επίσης να γράψει νέα σύμβολα. Στο παράδειγμά μας, ο κάστορας έχει το ρόλο της κεφαλής ανάγνωσης-εγγραφής.
- Ένα πεπερασμένο σύνολο καταστάσεων (*states*). Η συμπεριφορά της κεφαλής ανάγνωσης-εγγραφής μπορεί να αλλάζει ανάλογα με την κατάσταση. Στην περίπτωση μας, υπάρχουν μόνο δύο καταστάσεις, δηλαδή «κάστορας που κουβαλάει μπίλια» και «κάστορας που δεν κουβαλάει μπίλια».
- Ένα σύνολο κανόνων: Τι θα συμβεί, σε σχέση με την κατάσταση, όταν ένα συγκεκριμένο σύμβολο διαβαστεί από την ταινία; Πιθανές ενέργειες είναι: αλλαγή κατάστασης, εγγραφή νέου συμβόλου στην ταινία και μετακίνηση της κεφαλής ανάγνωσης/εγγραφής αριστερά ή δεξιά κατά ένα κελί.

Λέξεις κλειδιά και Websites

- Μηχανή Turing, (Turing machine):
https://en.wikipedia.org/wiki/Turing_machine
<https://de.wikipedia.org/wiki/Turingmaschine>

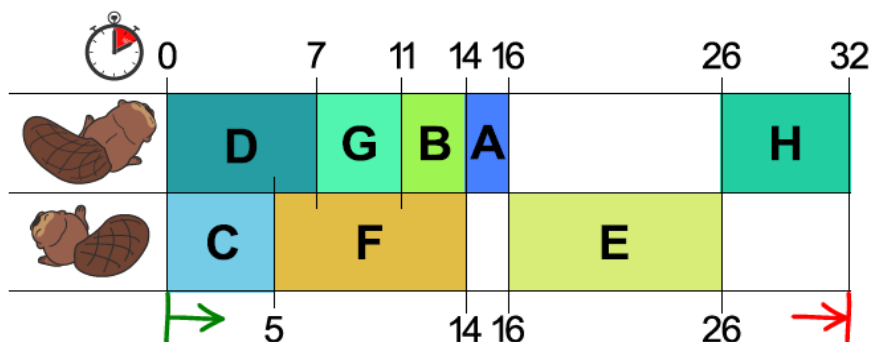
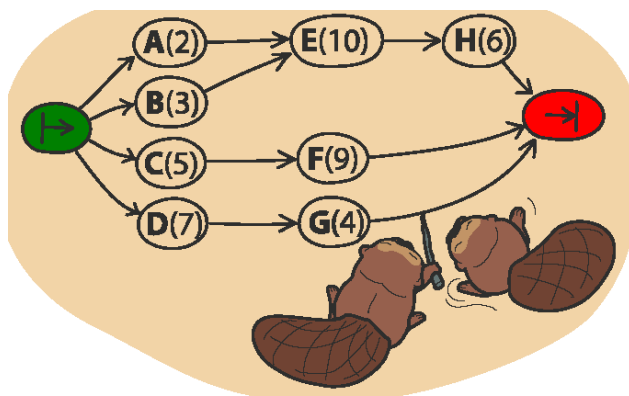




28. Καστορο-projects

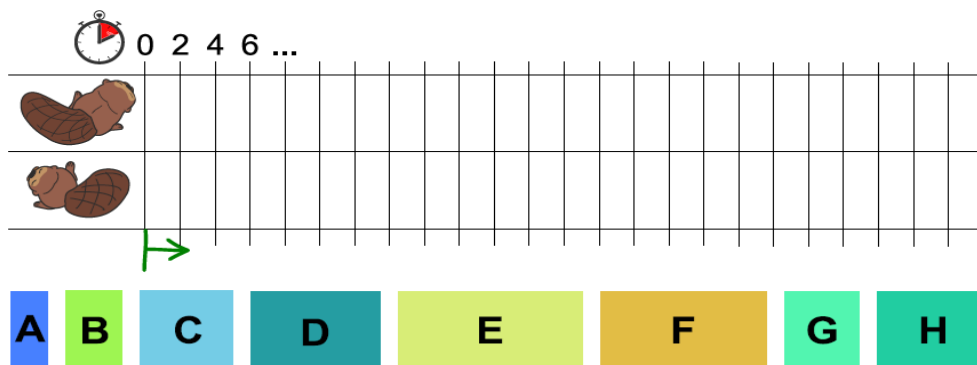
Η κατασκευή ενός φράγματος για κάστορες μπορεί να αναλυθεί σε διάφορες δευτερεύουσες εργασίες (κοπή δέντρων, αφαίρεση κλαδιών, μεταφορά κορμών στο νερό, κ.λπ.). Η εικόνα στα δεξιά τα δείχνει όλες τις 8 υποεργασίες Α, Β, C, D, E, F, G, H, καθεμία με τον αριθμό των ωρών που χρειάζεται για να τις ολοκληρώσει κανείς. Οι δευτερεύουσες εργασίες δεν είναι εντελώς ανεξάρτητες μεταξύ τους: Ένα βέλος από το Χ στο Υ σημαίνει ότι η δευτερεύουσα εργασία Χ πρέπει να ολοκληρωθεί πριν ξεκινήσει η δευτερεύουσα εργασία Υ.

Η κάστορας Ulla θέλει να βοηθήσει τον κάστορα Otso να χτίσει το φράγμα πιο γρήγορα. Μοιράζουν τις δευτερεύουσες εργασίες μεταξύ τους και δημιουργούν το ακόλουθο σχέδιο εργασίας που λαμβάνει υπόψη τις σχέσεις από την παραπάνω εικόνα



Με αυτό τον τρόπο το φράγμα θα είναι έτοιμο σε 32 ώρες. Όμως γίνεται και πιο γρήγορα!

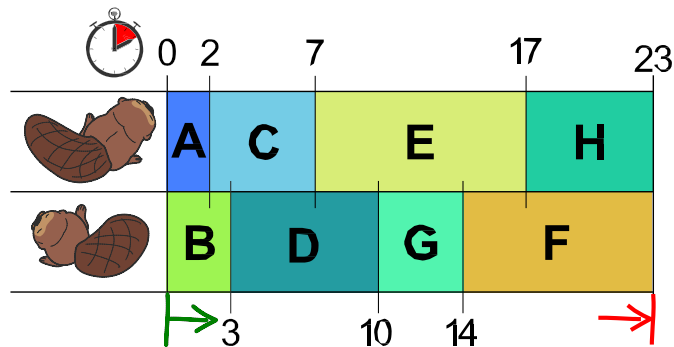
Δημιουργήστε ένα πρόγραμμα εργασίας για να ολοκληρώσετε το φράγμα σε όσο το δυνατόν λιγότερο χρόνο.



Λύση

Στο πρόγραμμα εργασίας που δίνεται στην άσκηση, ο πρώτος κάστορας έχει μεγάλο διάλειμμα (10 ώρες) και ο δεύτερος κάστορας έχει συνολικά 8 ώρες αδράνειας. Οι δύο θα τελείωναν πιο γρήγορα αν δούλευαν συνεχώς.

Ένα συνολικό ταχύτερο πρόγραμμα εργασίας επιτυγχάνεται εάν κάποιος βεβαιωθεί ότι οι δύο μεγαλύτερες εργασίες E(10) και F(9) δεν εκτελούνται από τον ίδιο κάστορα. Ακολουθεί ένα πιθανό πρόγραμμα εργασίας που ολοκληρώνεται σε 23 ώρες. Δεν θα μπορούσε να είναι πιο γρήγορο, γιατί οι δύο κάστορες δουλεύουν χωρίς διάλειμμα.



Αυτό είναι Πληροφορική!

Οι άνθρωποι είναι ανυπόμονοι, και ως αποτέλεσμα, υπάρχει συχνά η απαίτηση μια εργασία να γίνεται όσο το δυνατόν γρηγορότερα. Εάν η εργασία μπορεί να κατανεμηθεί (είτε σε πολλά άτομα, μηχανές ή κάστορες), τότε η κατανομή των δευτερευουσών εργασιών μεταξύ αυτών που επιτελούν την εργασία παίζει σημαντικό ρόλο στην ταχύτητα. Με τον ίδιο τρόπο, οι υπολογιστές πρέπει συχνά να μοιράζουν επιδέξια τις εργασίες υπολογισμού μεταξύ διαφορετικών πυρήνων ενός επεξεργαστή.

Υπάρχουν πολλές διαφορετικές στρατηγικές για τέτοια προβλήματα *χρονοπρογραμματισμού* (*scheduling problems*) που έχουν μελετηθεί ευρέως από την επιστήμη των υπολογιστών. Το πρώτο σχέδιο εργασίας σε αυτό το πρότζεκτ δημιουργήθηκε με τέτοιο τρόπο ώστε, μεταξύ των εργασιών που έπρεπε να γίνουν, αυτή με τη μεγαλύτερη διάρκεια ανατέθηκε σε έναν κάστορα που καθόταν - σε αυτήν την περίπτωση μια κακή στρατηγική. Συχνά λειτουργεί καλύτερα εάν ολοκληρώνονται πρώτα σύντομες εργασίες: η στρατηγική *Shortest-Job-Next* (SJN), γνωστή και ως *shortest job first* (SJF), «πρώτα η σύντομη δουλειά» ελαχιστοποιεί επίσης τον μέσο χρόνο αναμονής μέχρι την ολοκλήρωση κάθε εργασίας.

Λέξεις κλειδιά και Websites

- Χρονοπρογραμματισμός (Scheduling): <https://de.wikipedia.org/wiki/Scheduling>
- Shortest-Job-Next: <https://de.wikipedia.org/wiki/Shortest-Job-Next>



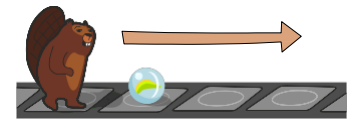
29. Αριθμο-μπίλιες

Οι κάστορες έχουν ένα ιδιαίτερο τρόπο να αναπαριστούν τους αριθμούς.



Τα διαφορετικά πεδία έχουν διαφορετικά βάρη και μια μπίλια στο αντίστοιχο τετράγωνο στο πεδίο δείχνει ότι η τιμή πρέπει να ληφθεί υπόψη. Στο παραπάνω παράδειγμα, αναπαρίσται ο αριθμός 52.

Ο κάστορας κινείται σε μια ταινία από αριστερά προς τα δεξιά, ανά ένα τετράγωνο κάθε φορά. Σε κάθε τετράγωνο της κορδέλας μπορεί να υπάρχει μία μπίλια.



Κάθε φορά που ο κάστορας πατάει σε ένα τετράγωνο και έχει τα χέρια του ελεύθερα, παίρνει τη μπίλια και μετά την κουβαλάει μαζί του.



Στο πρώτο ελεύθερο τετράγωνο αφήνει κάτω την μπίλια.



Ο κάστορας μπορεί να κουβαλήσει μόνο μία μπίλια και σε κάθε τετράγωνο χωράει μόνο μία μπίλια.

Όταν ο κάστορας κουβαλάει ήδη μία μπίλια και πατάει σε τετράγωνο που υπάρχει άλλη μπίλια

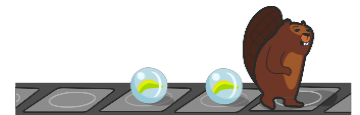


... τότε προσπερνάει το τετράγωνο.



... και αφήνει τη μπίλια του στο επόμενο ελεύθερο τετράγωνο

Μετά μπορεί να σηκώσει την επόμενη μπίλια.



ποιος αριθμός αναπαρίσται από τις μπίλιες, όταν ο κάστορας θα έχει διασχίσει την ταινία;

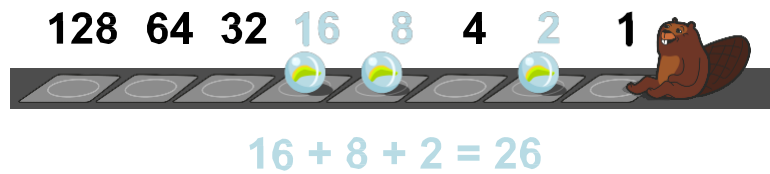


- A) 10 B) 26 C) 28 D) 104

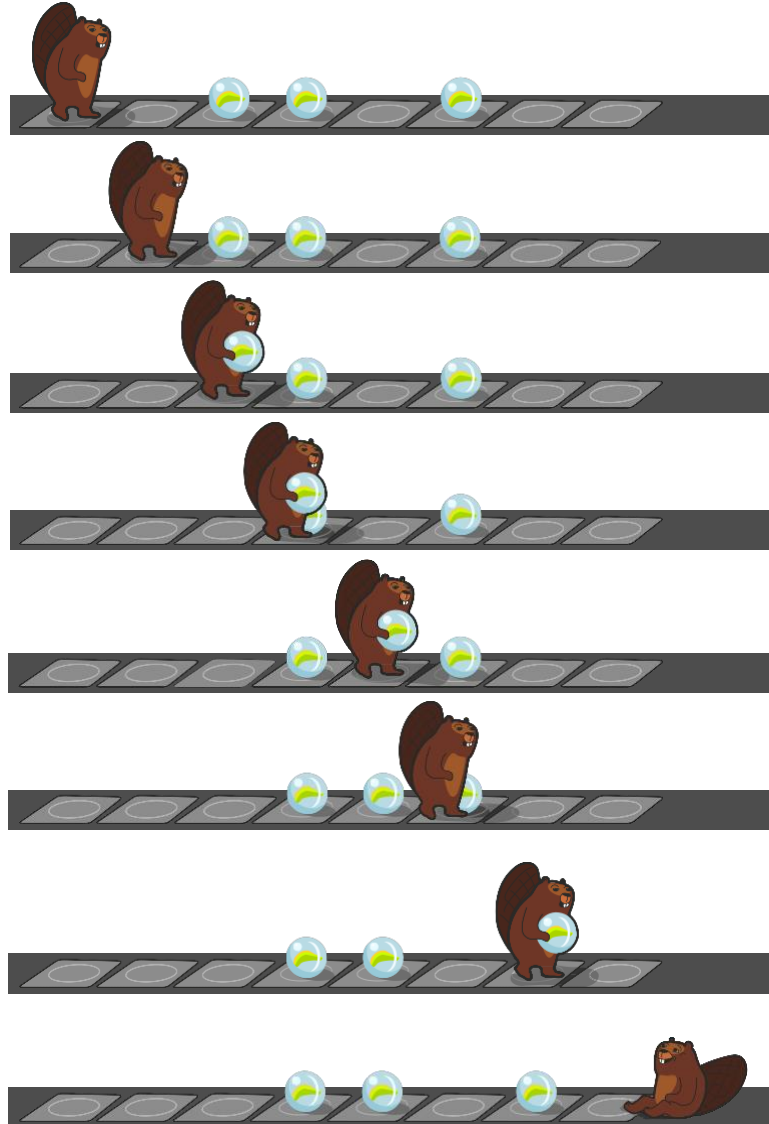


Λύση

Η σωστή απάντηση είναι η Β) 26



Το ακόλουθο γραφικό δείχνει τις ενέργειες:



Αυτό είναι Πληροφορική!

Στην επιστήμη των υπολογιστών, σχετικά απλές λειτουργίες συχνά οδηγούν σε ενδιαφέροντα αποτελέσματα. Αυτή η άσκηση είναι ένα καλό παράδειγμα. Ο κάστορας



ακολουθεί έναν αλγόριθμο (*algorithm*). Ο αλγόριθμος αυτός βασίζεται στο γεγονός ότι ο κάστορας μπορεί να έχει δύο διαφορετικές καταστάσεις (κουβαλάει μία μπίλια ή όχι) και ότι μπορεί να βρει δύο διαφορετικούς τύπους τετραγώνων στο δρόμο του (κατειλημμένο και άδειο).

Συνολικά, μετά την εκτέλεση του αλγόριθμου, το αποτέλεσμα είναι ακριβώς το ίδιο σαν να είχατε μετατοπίσει κάθε μπίλια στην ταινία μία θέση προς τα δεξιά. Στην αριθμητική αναπαράσταση του κάστορα, αυτό αντιστοιχεί σε διαίρεση με το 2. Εάν ο κάστορας διέσχιζε την ταινία από τα δεξιά προς τα αριστερά, ο αριθμός θα πολλαπλασιαζόταν επί 2. Εάν σε μια σειρά από 0 και 1, όλα τα στοιχεία μετατοπίζονται με τον ίδιο τρόπο προς τα αριστερά ή προς τα δεξιά, αυτό αναφέρεται ως *ολίσθηση των bit* (*bit shift*).

Όσο απλό κι αν είναι το παράδειγμα σε αυτό το πρόβλημα του κάστορα, περιέχει όμως μερικά από τα βασικά στοιχεία μιας *μηχανής Turing*. Μια μηχανή Turing (που πήρε το όνομά της από τον μαθηματικό Alan Turing) είναι ένας ειδικού σκοπού, πολύ απλά δομημένος υπολογιστής. Βασικά μια μηχανή Turing μπορεί να εκτελέσει οποιονδήποτε αλγόριθμο που μπορεί να εκτελέσει ένας συμβατικός υπολογιστής. Στην πράξη, ωστόσο, οι μηχανές Turing δεν χρησιμοποιούνται ως υπολογιστές γιατί μπορούμε να κατασκευάσουμε υπολογιστές που είναι ναι μεν πιο περίπλοκοι αλλά πολύ πιο αποτελεσματικοί. Οι μηχανές Turing είναι κυρίως χρήσιμες στη θεωρία. Λόγω της απλής δομής τους, είναι σχετικά εύκολο να αποδειχθούν θεωρίες με τις μηχανές Turing. Και αν μπορεί να αποδειχθεί ότι ένα πρόβλημα δεν μπορεί να λυθεί από τις μηχανές Turing, τότε δεν μπορεί να λυθεί από κανέναν υπολογιστή..

Μια μηχανή Turing αποτελείται από:

- Μία ταινία (*tape*) μεταβλητού μήκους που χωρίζεται σε κελιά (*cells*), το ένα δίπλα στο άλλο. Κάθε κελί περιέχει ένα *σύμβολο*. Στο παράδειγμά μας, τα κελιά είναι τα τετράγωνα στα οποία κινείται ο κάστορας.
- Ένα πεπερασμένο σύνολο *συμβόλων* (*symbols*). Συχνά χρησιμοποιούνται ως σύμβολα μόνο το 0 και το 1. Στο παράδειγμά μας η ύπαρξη μπίλιας αντιπροσωπεύει το 1 και ένα κενό τετράγωνο αντιπροσωπεύει το 0.
- Μια κεφαλή (*head*) ανάγνωσης/εγγραφής που κινείται και προς τις δύο κατευθύνσεις στην ταινία και, με αυτόν τον τρόπο, μπορεί να διαβάσει σύμβολα στην ταινία και επίσης να γράψει νέα σύμβολα. Στο παράδειγμά μας, ο κάστορας έχει το ρόλο της κεφαλής ανάγνωσης-εγγραφής.
- Ένα πεπερασμένο σύνολο *καταστάσεων* (*states*). Η συμπεριφορά της κεφαλής ανάγνωσης-εγγραφής μπορεί να αλλάζει ανάλογα με την κατάσταση. Στην περίπτωση μας, υπάρχουν μόνο δύο καταστάσεις, δηλαδή «κάστορας που κουβαλάει μπίλια» και «κάστορας που δεν κουβαλάει μπίλια».
- Ένα σύνολο κανόνων: Τι θα συμβεί, σε σχέση με την κατάσταση, όταν ένα συγκεκριμένο σύμβολο διαβαστεί από την ταινία; Πιθανές ενέργειες είναι: αλλαγή κατάστασης, εγγραφή νέου συμβόλου στην ταινία και μετακίνηση της κεφαλής ανάγνωσης/εγγραφής αριστερά ή δεξιά κατά ένα κελί.



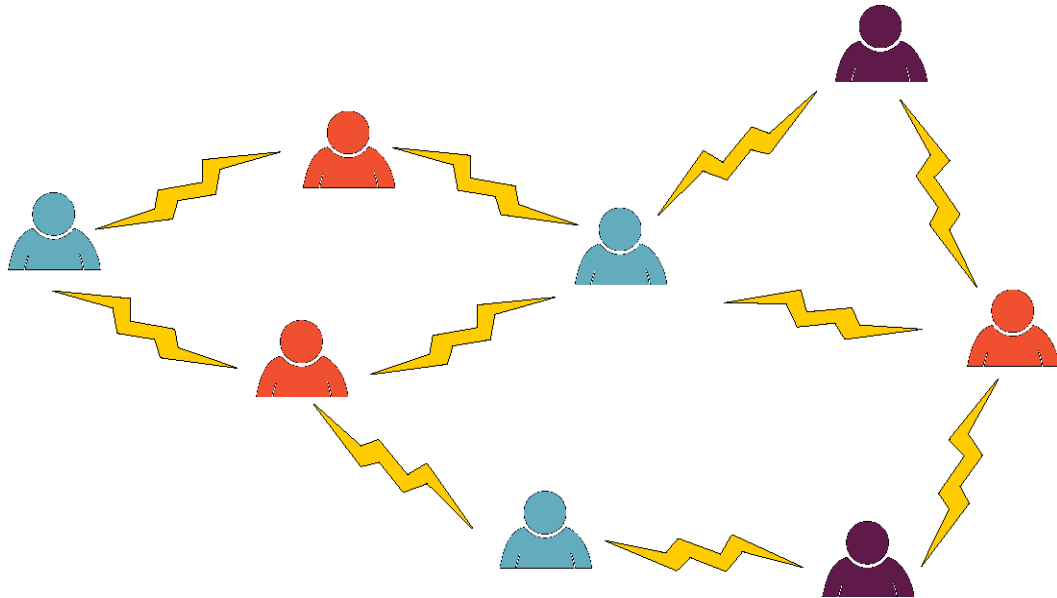
Λέξεις κλειδιά και Websites

- Μηχανή Turing, (Turing machine):
https://en.wikipedia.org/wiki/Turing_machine
<https://de.wikipedia.org/wiki/Turingmaschine>
- Λειτουργία σε επίπεδο bit, Bitwise operation:
https://en.wikipedia.org/wiki/Bitwise_operation#Bit_shifts



30. Teamwork

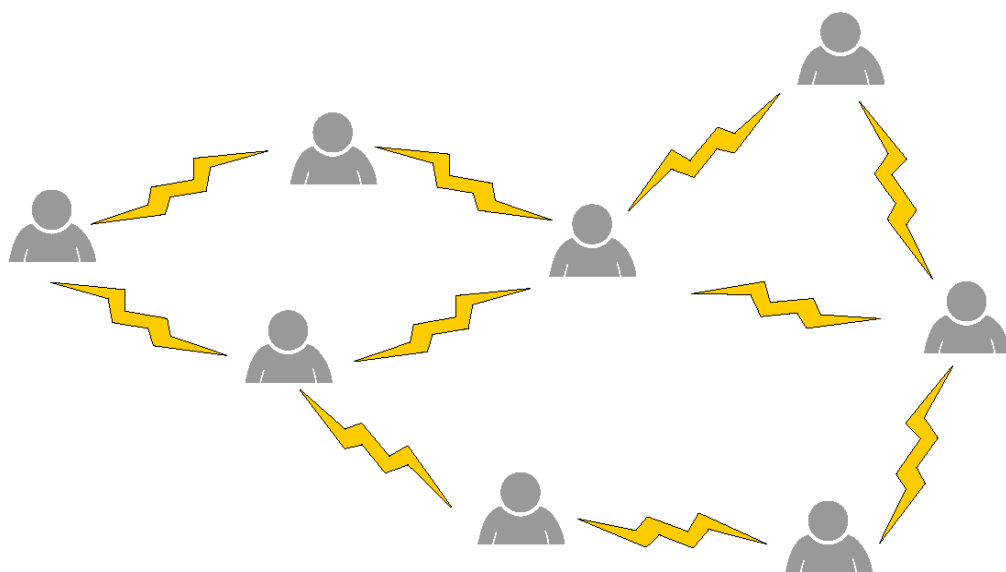
Για ένα project θα πρέπει να χωρίσετε οκτώ άτομα σε ομάδες. Υπάρχει ένας κεραυνός ανάμεσα σε δύο ανθρώπους που δηλώνει ότι δεν θέλουν να συνεργαστούν. Τότε δεν θέλετε να τους βάλετε στην ίδια ομάδα.



Εάν ληφθούν υπόψη οι αντιπάθειες στο παραπάνω παράδειγμα, τότε η διαίρεση είναι δυνατή σε τρεις ομάδες (κόκκινο, μπλε, βιολετί). Έτσι δεν υπάρχει κεραυνός πουθενά ανάμεσα σε δύο ανθρώπους του ίδιου χρώματος.

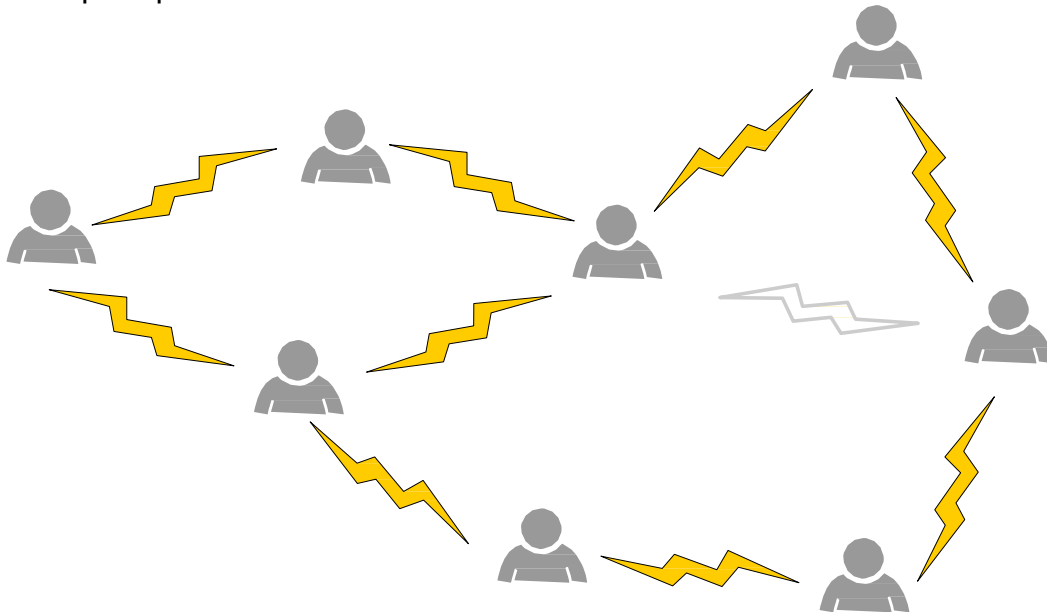
Εάν πείσετε τα κατάλληλα δύο άτομα να συνεργαστούν (δηλαδή αφαιρέσετε έναν κεραυνό), τότε είναι δυνατός ακόμη και ένας χωρισμός σε δύο μόνο ομάδες (μόνο δύο χρώματα).

Αφαίρεσε τον σωστό κεραυνό.



Λύση

Η σωστή λύση είναι:



Ας αναπαραστήσουμε την κατάσταση λίγο πιο αφηρημένα, με τους λεγόμενους γράφους (*graph*), με τους ανθρώπους ως κόμβους (*nodes*), είναι οι κύκλοι, και τους κεραυνούς ως ακμές (*edges*), είναι οι γραμμές που συνδέουν τους κόμβους.

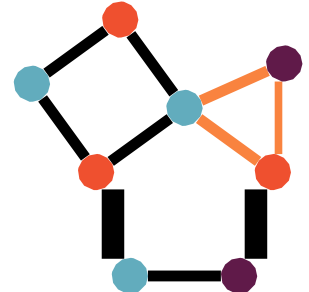
Η μοναδική δυνατότητα είναι η πορτοκαλί γραμμή.

Μετά το σβήσιμο αυτής της ακμής μπορούμε να χρωματίσουμε τους κόμβους με μόνο δύο χρώματα.

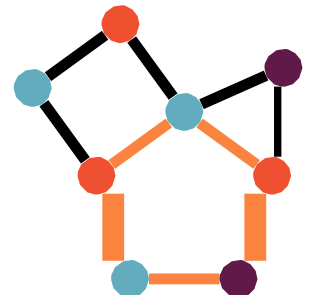


Κάθε χρώμα αντιπροσωπεύει μια ομάδα. Βλέπουμε ότι πουθενά δύο άτομα από την ίδια ομάδα δεν αρνούνται να συνεργαστούν: οι γειτονικοί κόμβοι έχουν παντού διαφορετικά χρώματα

Για να δούμε ότι το σβήσιμο αυτής της ακμής είναι η μοναδική λύση, ας παρατηρήσουμε το τρίγωνο με τις πορτοκαλί γραμμές



Εάν διαγραφεί μια γραμμή έξω από αυτό το τρίγωνο, χρειαζόμαστε ακόμα τρία χρώματα μόνο για τους τρεις κόμβους του τριγώνου.



Τώρα ας δούμε το πεντάγωνο με τις πορτοκαλί γραμμές:

Εάν μία γραμμή έξω από αυτό το πεντάγωνο διαγραφεί, αυτό παραμένει άθικτο και είναι αδύνατο να χρωματίσουμε το πεντάγωνο μόνο με δύο χρώματα: αν πάμε δεξιόστροφα μέσα από το πεντάγωνο, πρέπει να εναλλάσσουμε τα δύο χρώματα. Ωστόσο, όταν φτάσουμε στον τελευταίο κόμβο, θα έχει το ίδιο χρώμα με τον πρώτο κόμβο, αφού ο αριθμός των κύκλων στο πεντάγωνο είναι μονός, όπως και το τρίγωνο.

Άρα η μόνη δυνατή λύση είναι να διαγράψουμε την κοινή ακμή του τριγώνου και του πενταγώνου.

Αυτό είναι Πληροφορική!

Πολλά προβλήματα που εμφανίζονται στην καθημερινή ζωή μπορούν να διατυπωθούν ως προβλήματα χρωματισμού γράφων (*graph-coloring problem*). Στη συγκεκριμένη άσκησή μας, οι κόμβοι ενός γράφου αντιπροσωπεύουν τους ανθρώπους και μια ακμή μεταξύ δύο ατόμων δείχνει ότι αρνούνται να εργαστούν μαζί σε μια ομάδα. Αν χρωματίσουμε τους κόμβους με k χρώματα, αυτό μπορεί να γίνει κατανοητό ως αντιστοίχιση των προσώπων σε μία από τις k ομάδες. Ένας τέτοιος χρωματισμός ονομάζεται αποδεκτός εάν, για



οποιοδήποτε ζεύγος δύο κόμβων που συνδέονται απευθείας με μια ακμή, αυτοί οι δύο κόμβοι έχουν διαφορετικά χρώματα. Στην περίπτωση μας, ο χρωματισμός είναι επιτρεπτός εάν και μόνο εάν όλοι σε κάθε ομάδα συνεργάζονται. Μια ακμή καλείται *κρίσιμη* (*critical edge*) εάν η διαγραφή αυτής της ακμής επιτρέπει να χρωματιστεί ο γράφος με λιγότερα χρώματα από πριν. (Σε αυτή την περίπτωση οι κόμβοι του γράφου ενδέχεται να επαναχρωματιστούν.) Για εμάς, μια ακμή είναι τότε κρίσιμη, όταν τα αντίστοιχα δύο άτομα πειστούν να συνεργαστούν και έτσι ο γράφος μπορεί να χωριστεί σε λιγότερες ομάδες συνεργασίας.

Λέξεις κλειδιά και Websites

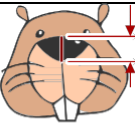
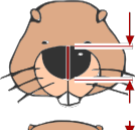
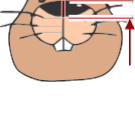
- Graphentheorie: <https://de.wikipedia.org/wiki/Graphentheorie>
- πρόβλημα χρωματισμού των γράφων (graph-coloring problem):
https://en.wikipedia.org/wiki/Graph_coloring
Färbeproblem: [https://de.wikipedia.org/wiki/Färbung_\(Graphentheorie\)](https://de.wikipedia.org/wiki/Färbung_(Graphentheorie))
- κρίσιμος γράφος (critical graph): https://en.wikipedia.org/wiki/Critical_graph
Kritischer Graph: https://de.wikipedia.org/wiki/Kritischer_Graph



31. Μέτρηση με νεύματα

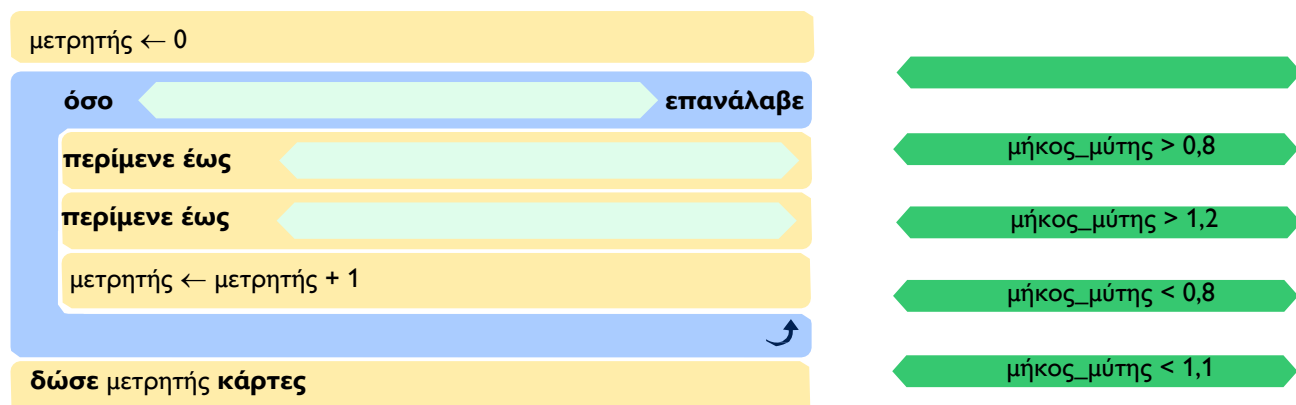
Ένα νέο μηχάνημα που εκδίδει εισιτήρια εισόδου θα πρέπει να λειτουργεί ως εξής: Ένας πελάτης κουνάει το κεφάλι του— δηλαδή χαμηλώνει το κεφάλι του και μετά κοιτάζει ευθεία – τόσες φορές όσες είναι τα εισιτήρια που θέλει να αγοράσει. Στο τέλος, ο πελάτης σηκώνει το κεφάλι του προς τα πάνω και στη συνέχεια το μηχάνημα δίνει τις κάρτες.

Για να γίνει αυτό το μηχάνημα έχει εγκατεστημένη κάμερα. Η κάμερα μπορεί να αναγνωρίσει τις μύτες των πελατών και να μετρά συνεχώς το μήκος της μύτης τους. Το πρόγραμμα ελέγχου του μηχανήματος αποθηκεύει το τρέχον αποτέλεσμα μέτρησης στη μεταβλητή «μήκος_μύτης» και διαφοροποιεί τις θέσεις του κεφαλιού του πελάτη με τη βοήθεια του παρακάτω πίνακα:

Μέτρηση κάμερας	Τιμή μεταβλητής μήκος_μύτης	Θέση του κεφαλιού
	1	Ο πελάτης κοιτάζει ευθεία
	1,3	Ο πελάτης έχει το κεφάλι κατεβασμένο
	0,7	Ο πελάτης έχει το κεφάλι ψηλά

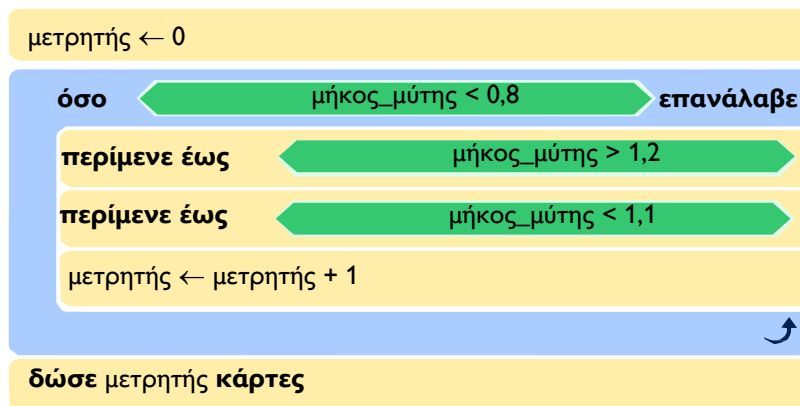
Το πρόγραμμα ελέγχου του μηχανήματος είναι σχεδόν έτοιμο, δες παρακάτω.

Ολοκλήρωσε το πρόγραμμα ελέγχου!

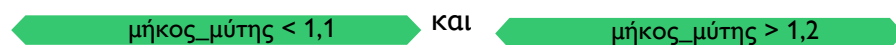


Λύση

Η μοναδική σωστή απάντηση είναι η:



Η δομή του προγράμματος είναι προκαθορισμένη: υπάρχει μια κεντρική εντολή επανάληψης. συχνά ονομάζεται βρόχος (*loop*). Η τελευταία από τις εντολές που επαναλαμβάνεται σε αυτόν τον βρόχο αυξάνει τον αριθμό των καρτών που πρέπει να δοθούν. Κατά συνέπεια, με τις δύο εντολές «περίμενε έως», γίνεται αντιληπτό ένα νεύμα του πελάτη: δηλ. ο πελάτης κατεβάζει πρώτα το κεφάλι του και μετά κοιτά ξανά ευθεία. Επομένως, η τιμή που αποθηκεύεται στη μεταβλητή μήκος_μύτης πρέπει πρώτα να πάει περίπου στο 1,3 και μετά να επιστρέψει στο 1. Αυτό αντιστοιχεί στις συνθήκες



Οι εντολές στην επανάληψη επαναλαμβάνονται έως ότου ο πελάτης σηκώσει το κεφάλι, δηλαδή όταν ανιχνευθεί μια τιμή η οποία είναι καθαρά μικρότερη του 1. Η συνθήκη που ταιριάζει καλύτερα είναι η $\text{μήκος_μύτης} < 0,8$

Ίσως έχετε παρατηρήσει ότι το πρόγραμμα δεν χρησιμοποιεί τις ακριβείς τιμές από τον πίνακα. Στην πράξη, δεν είναι δυνατή η συνεχής μέτρηση, αλλά μόνο σε μια συγκεκριμένη συχνότητα (π.χ. 25 φορές ανά δευτερόλεπτο). Μπορεί, για παράδειγμα, να μην μετρηθεί καθόλου η ακριβής τιμή για να κοιτάζουμε ευθεία μπροστά στο 1,0, επειδή μετρήθηκε πριν 0,95 και μετά 1,03.

Αυτό είναι Πληροφορική!

Η μηχανική όραση (*machine vision*, γνωστή και ως *computer vision*) είναι ένα πεδίο της επιστήμης των υπολογιστών στο οποίο σήμερα γίνεται εντατική έρευνα. Τόσο οι θεωρητικές συλλήψεις όσο και οι πρακτικές εφαρμογές έχουν μεγάλη σημασία.

Μια σημαντική εφαρμογή της μηχανικής όρασης είναι να δώσει στα άτομα με αναπηρία μια καλύτερη ευκαιρία να αλληλοεπιδρούν αυτόνομα με το περιβάλλον τους. Για παράδειγμα, ανάλογα με τη σοβαρότητα της αναπηρίας, ένα άτομο μπορεί να ελέγξει πολύ λίγους μύες. Ο παγκοσμίως φήμης φυσικός Stephen Hawking (1942–2018)





χρησιμοποίησε την κίνηση των μυών από τα μάγουλά του για να καθοδηγήσει ένα πρόγραμμα εκφοράς της φωνής, αφού είχε χάσει το μεγαλύτερο μέρος ελέγχου του μυϊκού του συστήματος.

Ωστόσο, το συγκεκριμένο παράδειγμα θα μπορούσε να χρησιμοποιηθεί και για τους μουσικούς: συνήθως χρησιμοποιούν και τα δύο χέρια για να χειριστούν το όργανό τους. Οι συσκευές που διατίθενται στο εμπόριο προσφέρουν ένα πεντάλ για το σκοπό αυτό. Ωστόσο, ορισμένοι μουσικοί, όπως οι οργανοπαίκτες χρησιμοποιούν επίσης τα πόδια τους για να παίξουν και θα μπορούσαν, για παράδειγμα, να γυρίζουν αυτόματα τις σελίδες με τις νότες με ένα απλό νεύμα.

Σε αντίθεση με το παράδειγμα αυτής της εργασίας, όπου οι τιμές είναι συγκεκριμένες και κωδικοποιημένες με σαφήνεια, η μηχανική όραση συχνά συνδυάζεται με τη μηχανική μάθηση (*machine learning*). Τότε το πρόγραμμα εκπαιδεύεται σε συγκεκριμένες χειρονομίες, με το να δέχεται πολλά παραδείγματα και αντιπαραδείγματα για κάθε χειρονομία. Με αυτόν τον τρόπο, το λογισμικό δημιουργεί μια στατιστική αξιολόγηση σχετικά με το πώς πρέπει να ερμηνευτεί κάτι.

Λέξεις κλειδιά και Websites

- Μηχανική όραση: (γερμανικά Maschinelles Sehen), Machine Vision, Computer Vision:
https://de.wikipedia.org/wiki/Computer_Vision
- Μηχανική μάθηση: Maschinelles Lernen, Machine Learning:
https://en.wikipedia.org/wiki/Machine_learning
- Stephen Hawking: https://en.wikipedia.org/wiki/Stephen_Hawking#Disability
- συσκευή που γυρνάει σελίδες, page turner, Notenwender:
<https://de.wikipedia.org/wiki/Notenwender>



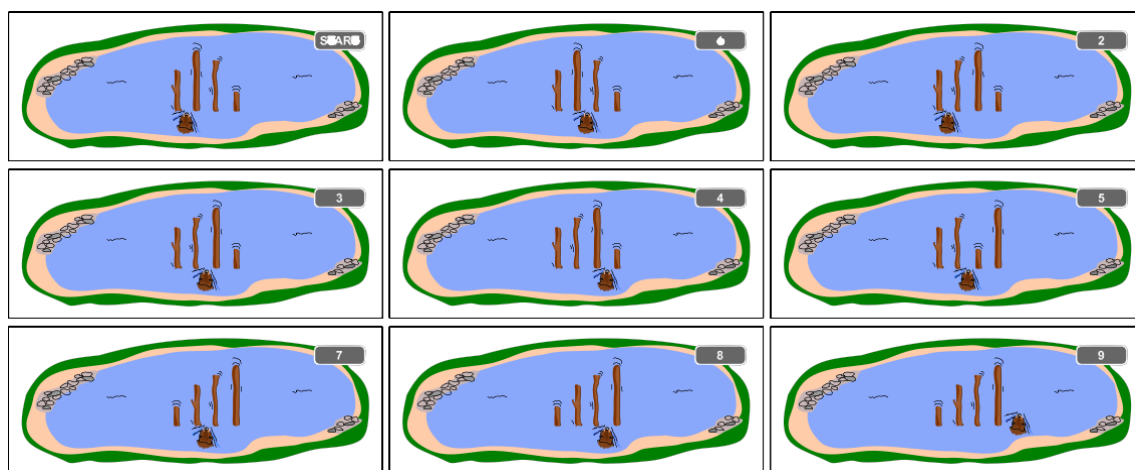


32. Ταξινόμηση του κάστορα

Ο κάστορας Hamid ταξινομεί τους κορμούς των δέντρων στη λίμνη. Αριστερά οι πιο κοντοί, δεξιά οι πιο μακριοί.

- Η αρχική θέση του Hamid είναι μεταξύ των δύο κορμών, τέρμα αριστερά.
- Αν είναι ανάμεσα σε δύο γειτονικούς κορμούς, τους συγκρίνει:
 - Αν ο δεξιός κορμός είναι μακρύτερος από τον αριστερό, ο κάστορας θα κολυμπήσει μία θέση προς τα δεξιά
 - Αν, από την άλλη, ο αριστερός κορμός είναι πιο μακρύς, τότε αλλάζει τους δύο κορμούς. Μετά την εναλλαγή, αν είναι στην αρχική θέση κολυμπάει μία θέση προς τα δεξιά, αλλιώς μια θέση προς τα αριστερά.
- Ο Hamid συνεχίζει με αυτόν τον τρόπο μέχρι να φτάσει στα δεξιά όλων των κορμών. Τότε ξέρει, ότι όλοι οι κορμοί είναι σωστά ταξινομημένοι

Το παράδειγμα δείχνει πως ταξινομεί ο Hamid τους 4 κορμούς. Συνολικά κάνει 9 συγκρίσεις.



Ο αριθμός των συγκρίσεων εξαρτάται από τη θέση των κορμών στην αρχή. Για 4 κορμούς ο Hamid πρέπει να κάνει τουλάχιστον 3 συγκρίσεις (αν τα στελέχη είναι ήδη σωστά ταξινομημένα) και το πολύ 9 συγκρίσεις (όταν κανένας κορμός δεν είναι στη σωστή θέση). Με 4 κορμούς δέντρων, ο Hamid πρέπει να υπολογίσει με 3 έως 9 συγκρίσεις.

Ο Χαμίντ τώρα πρέπει να ταξινομήσει 40 κορμούς δέντρων διαφορετικού μήκους. Πόσες συγκρίσεις μπορεί να περιμένει;

- A) 0 έως 20 συγκρίσεις
- B) 3 έως 40 συγκρίσεις
- C) 39 έως 120 συγκρίσεις
- D) 39 έως 1521 συγκρίσεις



Λύση

Στην καλύτερη περίπτωση, ο Hamid χρειάζεται 39 συγκρίσεις για 40 κορμούς, δηλαδή όταν όλοι οι κορμοί δέντρων έχουν ήδη ταξινομηθεί και επομένως κολυμπάει πάντα προς τα δεξιά μέχρι να τελειώσει. Επομένως, οι απαντήσεις A) και B) δεν μπορούν να είναι σωστές.

Στη χειρότερη περίπτωση, οι κορμοί των δέντρων είναι σε εντελώς διαφορετική θέση από ότι εάν ήταν ταξινομημένοι. Εδώ ο αριθμός των συγκρίσεων είναι πιο δύσκολο να προσδιοριστεί. Αρχικά παρατηρούμε τα εξής: Οι κορμοί των δέντρων στα αριστερά του Hamid είναι πάντα σωστά ταξινομημένοι μεταξύ τους. Όταν ο Hamid φτάνει σε έναν κορμό πιο κοντό από τους k κορμούς στα αριστερά του, τον αλλάζει πρώτα προς τα αριστερά, κάνοντας k συγκρίσεις. Μετά επιστρέφει και κολυμπάει ακόμα ένα προς τα δεξιά ενώ κάνει $k - 1$ συγκρίσεις, αλλά δεν ανταλλάσσει κανέναν κορμό πια. Συνολικά αυτό είναι $k + (k - 1) = 2k - 1$ συγκρίσεις. Αυτό συμβαίνει σε κάθε μία από τις $n - 1$ θέσεις μεταξύ δύο κορμών, ενώ ο αριθμός των κορμών στα αριστερά του Hamid είναι κατά μέσο όρο $\frac{n}{2}$. Έτσι το σύνολο των συγκρίσεων είναι

$(n - 1) \cdot (2 \cdot \frac{n}{2} - 1) = (n - 1)^2$, δηλαδή $39^2 = 1521$ συγκρίσεις. Έτσι η απάντηση C) με 120 το πολύ συγκρίσεις είναι λάθος. Σωστή απάντηση είναι η D).

Αυτό είναι Πληροφορική!

Η ταξινόμηση στοιχείων είναι μια κλασική εργασία στην επιστήμη των υπολογιστών. Η αποτελεσματική ταξινόμηση εξοικονομεί πολύ χρόνο. Ο αλγόριθμος ταξινόμησης που παρουσιάζεται εδώ ονομάζεται *Gnome Sort* και αρχικά παρουσιάστηκε ως *Stupid Sort* από τον Ιρανό επιστήμονα υπολογιστών Hamid Sarbazi-Azad. (Τα garden gnomes, γερμανικά: Gartenzwerge, «νάνοι του κήπου», είναι διακοσμητικά ειδώλια για τον κήπο. Παραπέμπουν στους καλικάντζαρους και συναντώνται συχνά στους κήπους των σπιτιών στην Ευρώπη). Αργότερα, ο Ολλανδός επιστήμονας υπολογιστών Dick Grune ονόμασε αυτή τη μέθοδο ταξινόμησης *Gnome Sort*, με την ιδέα ότι μόνο ένα garden gnome θα μπορούσε να ταξινομήσει τις γλάστρες με τα λουλούδια με αυτόν τον τρόπο.

Η ανάλυση του χρόνου εκτέλεσης των αλγορίθμων (δηλαδή η εύρεση του χρόνου που μπορεί να χρειαστεί για να ολοκληρωθεί ένας αλγόριθμος) είναι πολύ σημαντική στην επιστήμη των υπολογιστών. Συχνά ρωτά κανείς ποιος είναι ο χρόνος εκτέλεσης στην καλύτερη περίπτωση, ο μέσος όρος και η χειρότερη περίπτωση, ανάλογα με το μέγεθος εισόδου n . (Στην περίπτωση μας χρησιμοποιούμε τον αριθμό των κορμών δέντρων για αυτό.) Για να κάνουμε τη σύγκριση μεταξύ των διαφορετικών χρόνων λειτουργίας που απαιτούνται όσο το δυνατόν πιο απλή, συνήθως αρκούμαστε στο να τους δίνουμε μόνο κατά προσέγγιση. Στην καλύτερη περίπτωση, το *Gnome Sort* χρειάζεται μόνο γραμμικό χρόνο εκτέλεσης – η πολυπλοκότητά του είναι γραμμική και συμβολίζεται ως $O(n)$ – στη μέση περίπτωση χρειάζεται τετραγωνικό χρόνο εκτέλεσης – η πολυπλοκότητά του είναι



τετραγωνική και συμβολίζεται ως $O(n^2)$ - και στη χειρότερη περίπτωση χρειάζεται επίσης τετραγωνικός χρόνος εκτέλεσης – επίσης $O(n^2)$.

Λέξεις κλειδιά και Websites

- Gnome Sort: <https://de.wikipedia.org/wiki/Gnomesort>
- Sortiervverfahren: <https://de.wikipedia.org/wiki/Sortiervverfahren>
- Laufzeitanalyse
- O-Notation, Landau-Symbole: <https://de.wikipedia.org/wiki/Landau-Symbole>





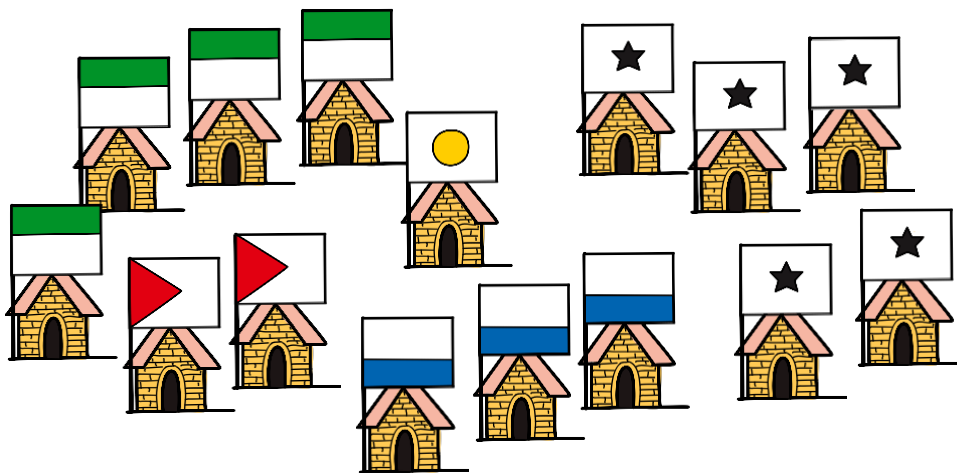


33. Οι οικογένειες της Καστορίας

Η Καστορία φιλοξενεί πέντε κάποτε εχθρικές μεταξύ τους οικογένειες (clans), η καθεμία με λίγα σπίτια, όπως φαίνεται στην εικόνα: οι Mac Intosh, οι Applers, οι Mac Rosofts, οι Androidians και οι Freefolders. Δεδομένου ότι οι καιροί είναι από καιρό ειρηνικοί, αποφασίζουν να πραγματοποιήσουν το τελετουργικό της συμφιλίωσης. Οι κανόνες για αυτό είναι:

- Μόνο δύο οικογένειες μπορούν να συμφιλιωθούν ταυτόχρονα.
- Σε κάθε σπίτι θα πραγματοποιηθεί μια εβδομαδιαία γιορτή για να σηματοδοτήσει το σύμφωνο συμφιλίωσης. Η διάρκεια λοιπόν της συμφιλίωσης σε εβδομάδες είναι επομένως ίση με τον αριθμό των σπιτιών στις δύο οικογένειες.
- Μετά από αυτό το διάστημα, οι δύο οικογένειες είναι μόνο μία. Η συνένωση των οικογενειών μπορεί να συνεχιστεί.

Οι οικογένειες αποφασίζουν να πραγματοποιήσουν την συνένωση στο συντομότερο δυνατό χρόνο. Αυτό μπορεί να γίνει μόνο με προσεκτικό σχεδιασμό της σειράς των συνενώσεων.



Πόσες εβδομάδες θα χρειαστούν έτσι ώστε όλες οι οικογένειες να συνενωθούν;

- A) 15 εβδομάδες
- B) 33 εβδομάδες
- C) 35 εβδομάδες
- D) 50 εβδομάδες
- E) 120 εβδομάδες

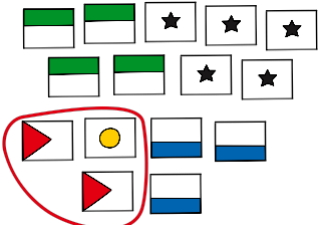
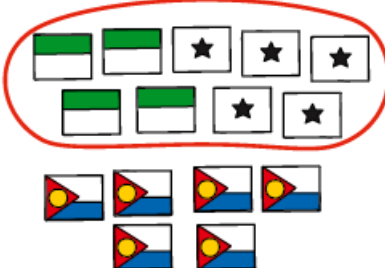
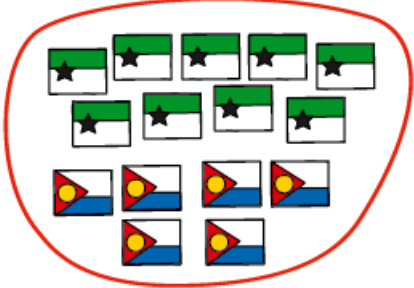


Λύση

Η σωστή απάντηση είναι Β) 33 εβδομάδες.

Η βέλτιστη στρατηγική για την ελαχιστοποίηση του συνολικού αριθμού εβδομάδων που χρειάζονται για να συνενωθούν όλες οι φυλές είναι να ελαχιστοποιηθεί ο αριθμός των σπιτιών που εμπλέκονται κάθε φορά. Τα σπίτια των συμμοριών που συνενώνονται πρώτες λαμβάνονται επίσης υπόψη στις περαιτέρω συνενώσεις. Επομένως, είναι λογικό να ενωθούν πρώτα οι μικρότερες οικογένειες, ώστε οι μεγάλες να μην χρειάζεται να συμμετέχουν πολύ συχνά στην τελετή. Για να επιτευχθεί αυτό, θα πρέπει σε κάθε βήμα να συνενωθούν οι δύο οικογένειες με τα λιγότερα σπίτια.

Αυτή η βήμα προς βήμα διαδικασία φαίνεται στον παρακάτω πίνακα, παραλείπουμε τα ονόματα των φυλών για λόγους σαφήνειας και αναφέρουμε μόνο τα μεγέθη των φυλών σε παρενθέσεις

	(1) και (2) συνενώνονται στην οικογένεια (3), χρειάζονται 3 εβδομάδες
	(3) και (3) συνενώνονται στην οικογένεια (6) σε 6 εβδομάδες.
	Οι (4) und (5) σε 9 εβδομάδες συνενώνονται στην (9).
	Στο τέλος συνενώνονται οι (6) και (9). Αυτό απαιτεί 15 εβδομάδες.





Συνολικά η διαδικασία ολοκληρώνεται μετά από $15 + 9 + 6 + 3 = 33$ εβδομάδες, σωστή είναι η απάντηση Β).



Αυτό είναι Πληροφορική!

Αυτό είναι ένα πρόβλημα βελτιστοποίησης (*optimization problem*), μια εργασία ελαχιστοποίησης ή μεγιστοποίησης μιας τιμής (σε αυτή την περίπτωση η διάρκεια της συνένωσης). Για αυτό χρησιμοποιούνται συχνά αλγόριθμοι (*algorithms*), οι οποίοι σήμερα είναι πλέον απαραίτητοι, για παράδειγμα, για την υλοποίηση διαδικασιών παραγωγής που εξοικονομούν πόρους και είναι φιλικές προς το περιβάλλον.

Σε αυτή την περίπτωση, το πρόβλημα μπορεί να λυθεί βέλτιστα ακόμη και με έναν απλό άπληστο αλγόριθμο (*greedy algorithm*). Εδώ κάνει κανείς σε κάθε βήμα αυτό που φαίνεται να είναι καλύτερο, βασιζόμενος σε μια απλή ιδέα. Στην περίπτωσή μας, η ιδέα είναι ότι πρώτα ενώνουμε μικρές οικογένειες γιατί χρειάζεται ο λιγότερος χρόνος.

Λέξεις κλειδιά και Websites

- Πρόβλημα βελτιστοποίησης (Optimization problem, Optimierungsproblem)
https://en.wikipedia.org/wiki/Optimization_problem
<https://de.wikipedia.org/wiki/Optimierungsproblem>
- άπληστος αλγόριθμος (Greedy-Algorithm)
<https://de.wikipedia.org/wiki/Greedy-Algorithmus>
https://en.wikipedia.org/wiki/Greedy_algorithm



A. Συγγραφείς των ασκήσεων

 Daumilas Ardickas	 Ezgi Arzu Güneş
 Sarah Atkins	 Mathias Hiron
 Michael Barot	 Benjamin Hirsch
 Liam Baumann	 Juraj Hromkovič
 Wilfried Baumann	 Andrea Hrušecká
 Linda Björk Bergsveinsdóttir	 Alisher Ikramov
 Javier Bilbao	 Tiberiu Iorgulescu
 Lucia Budinská	 Svetlana Jaksic
 Carmel Carroll	 YongJu Jeon
 Sarah Chan	 Soojin Jun
 Anton Chukhnov	 Ungyeol Jung
 Kris Coolsaet	 Filiz Kalelioğlu
 Valentina Dagienė	 Martin Kandlhofer
 Christian Datzko	 Ulrich Kiesmüller
 Susanne Datzko	 Dong Yoon Kim
 Janez Demšar	 Jihye Kim
 Nora A. Escherle	 Vaidotas Kinčius
 Lidia Feklistova	 V́ictor Koleszar
 Margarita Flores-Sicich	 Regula Lacher
 Fabian Frei	 Taina Lehtimäki
 Gerald Futschek	 Marielle Léonard
 Jens Gallenbacher	 Angélica Herrera Loyo
 Thomas Galler	 Tom Naughton
 Mark Edward M. Gonzales	 Mochammad Irfan Noviana
 Martin Guggisberg	 Graciela Oyhenard
 Yasemin Gülbahar	 Gabriela Lourdes Rodríguez Parada



A. Συγγραφείς των ασκήσεων

 Jean-Philippe Pellet

 Hannah Piper

 Jonatan Pipping

 Zsuzsa Pluhár

 Wolfgang Pohl

 Peter Rossmanith

 Rodrigo Santamaría

 Eljakim Schrijvers

 Rosario Schunk

 Tomas Šiaulys

 Timur Sitdikov

 Bernadette Spieler

 Cuttle.org Team

 Ezra Templonuevo

 Ahto Truu

 Troy Vasiga

 Florentina Voboril

 Eslam Wageed

 Michael Weigend

 Kyra Willekes

 Hongjin Yeh

 Mija Zaļūksne





B. Sponsoring: Wettbewerb 2021

HASLERSTIFTUNG

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

Stiftungszweck der Hasler Stiftung ist die Förderung der Informations- und Kommunikationstechnologie (IKT) zum Wohl und Nutzen des Denk- und Arbeitsplatzes 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: «Haus- gemacht 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 Erprobens 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

Scuola universitaria professionale
della Svizzera italiana



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

La Scuola universitaria professionale della Svizzera italiana
(SUPSI)



PADAGOGISCHE
HOCHSCHULE
ZÜRICH



<https://www.phzh.ch/>
Pädagogische Hochschule Zürich





C. Weiterführende Angebote

Das Lehrmittel zum Informatik-Biber

Module

Verkehr – Optimieren

Musik – Komprimieren

Geheime Botschaften – Verschlüsseln

Internet – Routing

Apps

Auszeichnungssprachen

<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 LernNetz 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.

010100110101011001001001
010000010010110101010011
010100110100100101000101
001011010101001101010011
010010010100100100100001

SV!A

www.svia-ssie-ssii.ch
schweizerischerverein für informatik in der
ausbildung // société suisse pour l'infor-
matique dans l'enseignement // société sviz-
zeraperl'informaticanell'insegnamento

Werden Sie SVIA Mitglied – <http://svia-ssie-ssii.ch/>

[svia/mitgliedschaft](http://svia-ssie-ssii.ch/mitgliedschaft) und unterstützen 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.

