ISSN 2783-7599

b·e·b·r·a·s

Informatikos ir informatinio mąstymo uždavinių rinkinys

Nr. 7

ŠVIETIMO, MOKSLO IR SPORTO MINISTERIJA

VILNIAUS UNIVERSITETAS · 1579 · UNIVERSITAS VILNENSIS

Bebras
International Challenge on Informatics and Computational Thinking

Šiame rinkinyje pateikiami 2020 metų XVII informatikos ir informatinio mąstymo konkurso (iššūkio) „Bebras“ I etapo uždaviniai, jų atsakymai ir paaiškinimai, koks informatikos turinys ir konceptai atskleidžiami, kaip ir kuo uždavinys įdomus ugdant informatinį mąstymą. Visi uždaviniai (įskaitant grafiką ir kitą medžiagą) licencijuojami pagal Kūrybinių bendrijų licenciją – „Creative Commons Attribution-ShareAlike 4.0 International License“. Šis uždavinių rinkinys skiriamas 1–12 klasių mokinių informatinio mąstymo gebėjimams ugdyti.

Dėkojame Daumilui Ardickui, dr. Gintautui Grigui, dr. Tatjanai Jevsikovai, dr. Eimučiui Karčiauskui, Alvidai Lozdienei, Modestui Rimkui, dr. Gabrielei Stupurienei, Tomui Šiauliui, dr. Sigitai Turskienei, talkinusiems verčiant ir adaptuojant uždavinius. Taip pat dėkojame tarptautinei „Bebro“ bendruomenei ir uždavinių autoriams.

Parengė Lina Vinikienė

Konsultavo Valentina Dagienė

Redagavo Viktoras Dagys

Viršelį kūrė Vaidotas Kinčius

Įvadas

Informatinis mąstymas – tai gebėjimas atpažinti, formuluoti ir spręsti aplinkos problemas (uždavinius), logiškai organizuoti ir analizuoti duomenis, taikyti schemas ir modelius, įvertinti problemos išsprendžiamumą, bandyti automatizuoti sprendimą naudojantis skaitmeninėmis technologijomis.

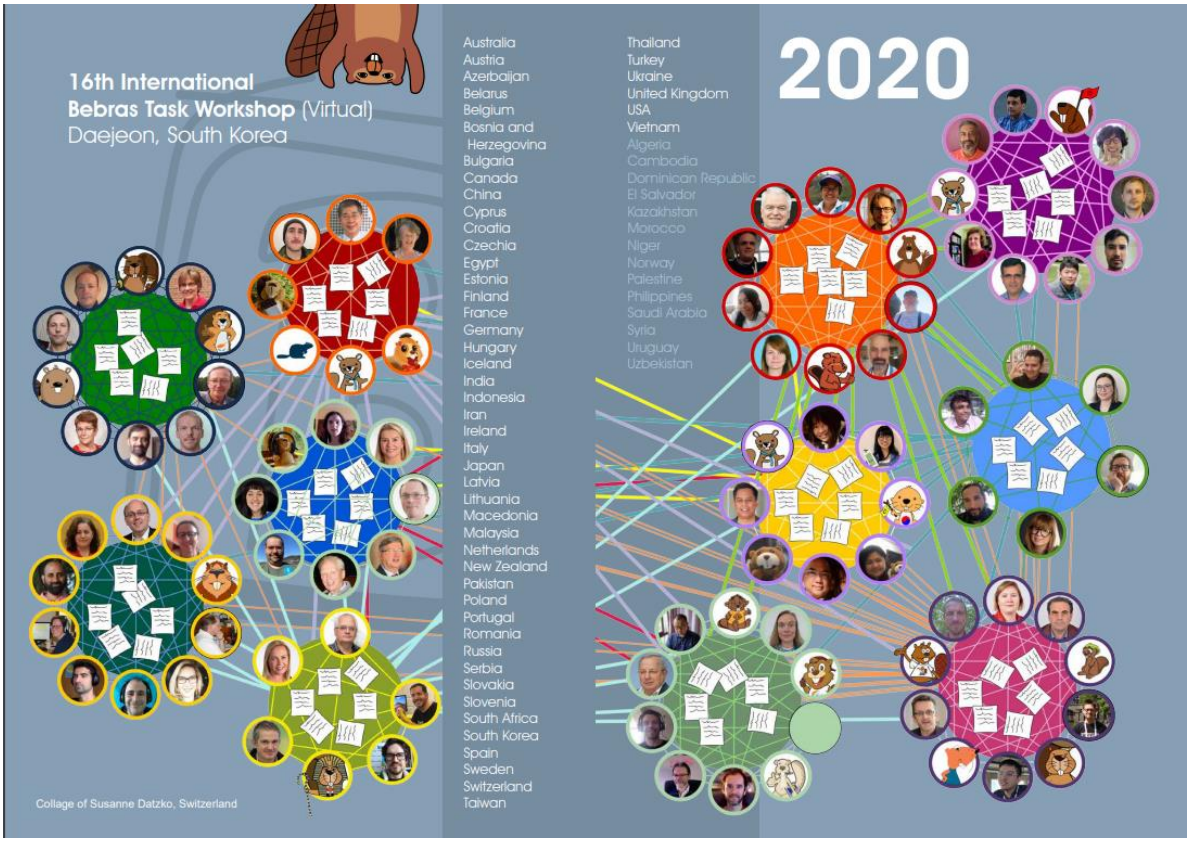
2006 metais informatinio mąstymo idėją ėmėsi aktyviai propaguoti Jeannette M. Wing (*Computational thinking. Communication of ACM, 49, 33–35*), tuo metu JAV Kolumbijos universiteto Duomenų mokslo instituto informatikos profesorė, viena iš „Microsoft Research“ vadovų. Tačiau pradinė informatinio mąstymo idėja priklauso daugeliui Lietuvos pedagogų žinomos knygos „Minčių audros: vaikai, kompiuteriai ir veiksmingos idėjos“ (*Mindstorms: Children, Computers, and Powerful Ideas*) autoriui Seimūrui Papertui (Seymour Papert). Jis jau prieš pusšimtį metų rašė, kad turime mokyti vaikus mąstyti apie savo mąstymą, gerinti jį pasitelkę įvairias priemones, taip pat ir kompiuterį, turime spręsti realaus gyvenimo uždavinius ir jų sprendimui pasinaudoti technologijomis.

„Bebras“ – tai ne tik konkursas (anglų kalba jis vadinamas *challenge* – iššūkiu). Tai daugybė įvairių veiklų, kurios vyksta ištisus metus. Susikūrė stiprus pasaulinis „Bebro“ mokslininkų ir mokytojų tinklas, kasmet kuriami nauji uždaviniai, ieškoma aktualių temų. Konkursui vykdyti reikalingos modernios sistemos, dalis šalių kuria ir tobulina šias sistemas pačios, kitos naudojasi kai kurių šalių paslaugomis.

Vykdant konkursą kaupiami mokinių sprendimai – daugybė duomenų surinkta, galima atlikti tyrimus, stebėti ir pan. Įvairių šalių mokslininkai, remdamiesi „Bebro“ bendruomenės sukauptais duomenimis, rengia ir publikuoja mokslinius straipsnius (žr. https://www.bebras.org/publications.html). Rengiamos parodos, festivaliai, kuriuose aptariami „Bebro“ uždaviniai. Daugelis šalių leidžia knygeles, kuriose aiškinami uždavinių sprendimai ir, svarbiausia, kaip kiekvienas uždavinys susijęs su informatika, kokie konceptai paslėpti, parodoma, kaip galima eiti toliau ir giliau.

Mokiniai, spręsdami įdomius, informatikos konceptais grįstus uždavinius, susipažįsta su informatikos sąvokomis, pagilina jų supratimą, išsiugdo gebėjimą taikyti jas praktiškai, uždavinių sprendimus aptaria su bendraamžiais ir mokytojais. Didelė dalis uždavinių yra iš algoritmų ir programavimo srities. Juos perpratus, skatinama mokytis praktinio programavimo. Mokytojams uždaviniai – didaktiniai ištekliai, idėjos savo pamokoms paįvairinti. Mokslininkai, informatikos tyrėjai, kurdami uždavinius, stengiasi perteikti esminius

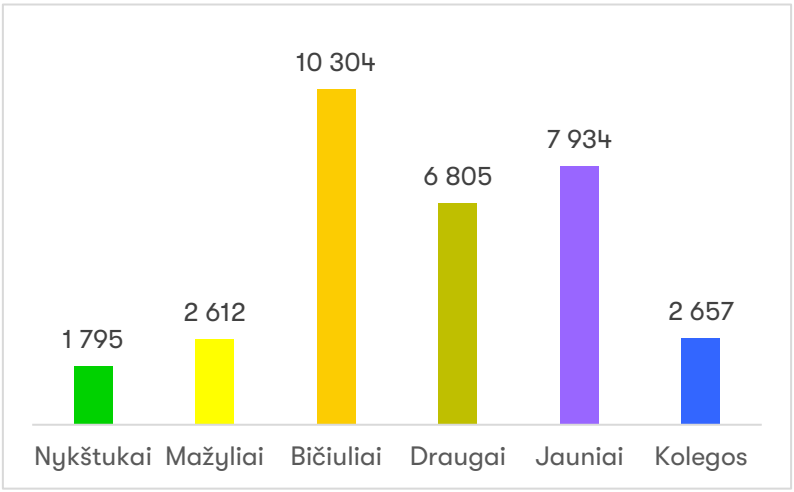
informatikos mokslo principus. „Bebro“ bendruomenę labiausiai vienija patrauklių, įdomių informatikos uždavinių paieška, informatikos konceptų išreiškimas žaidybinėmis užduotimis.



XVI tarptautinis „Bebro“ seminaras-dirbtuvės pirmą kartą vyko nuotoliniu būdu

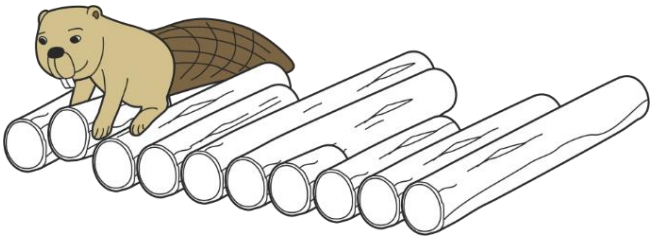
2020 metų „Bebro“ konkurso uždavinius sprendė 2 479 870 mokinių 57-iose valstybėse. Lietuvoje tais metais konkurse dalyvavo 32 107 mokiniai:

- 1 795 nykštukai (1–2 klasės)
- 2 612 mažyliai (3–4 klasės)
- 10 304 bičiuliai (5–6 klasės)
- 6 805 draugai (7–8 klasės)
- 7 934 jauniai (9–10 klasės)
- 2 657 kolegos (11–12 klasės)

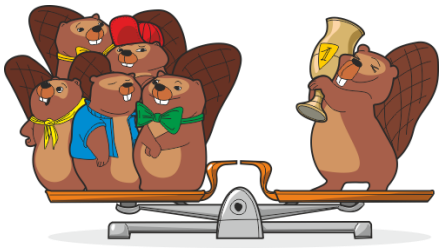


Konkurso dalyvių skaičius Lietuvoje 2020 m.

2020 metais Lietuvoje kiekviena amžiaus grupė sprendė po 18 uždavinių, išskyrus nykštukus (12 uždavinių) ir mažylius (15 uždavinių): trečdalis buvo lengvesnių, trečdalis – vidutinio sunkumo ir trečdalis – sunkesnių. Uždaviniams spręsti skiriamos 45 minutės.



Sutartas „Bebro“ uždavinių vertinimo modelis, tačiau šalys gali jį kiek modifikuoti. Lietuvoje taškai skaičiuojami taip:



- Prieš pradėdamas spręsti kiekvienas dalyvis turi 54 taškus (mažylių grupėje – 36 taškus)
  (18 uždavinių × 3 arba 12 uždavinių × 3);
- Už teisingai išspręstą uždavinį skiriama 6, 9 arba 12 taškų (priklausomai nuo uždavinio sunkumo lygio);
- Už neišspręstą uždavinį – 0 taškų;
- Už klaidingą atsakymą atimamas trečdalis uždaviniui skiriamų taškų, t. y., atitinkamai 2, 3 arba 4 taškai.

Daugiau informacijos apie „Bebro“ konkursą pateikiama interneto svetainėse:

- Tarptautinė „Bebro“ iššūkio svetainė: www.bebras.org
  (iš čia galima pasiekti visų dalyvaujančių šalių svetaines – spustelėkite šalies vėliavėlę)
- Lietuvos „Bebro“ svetainė: bebras.lt
- Konkurso sistema – vadinamasis „Bebro“ varžybų laukas: lt.bebras.lt

www.bebras.org

bebras.lt

lt.bebras.lt

Lentelėje pateikiamas šio rinkinio uždavinių skirstymas pagal amžiaus grupes. Skaičius langelyje prie uždavinio nurodo sudėtingumo lygį:

- lengvas – 6,
- vidutinis – 9,
- sunkus – 12.

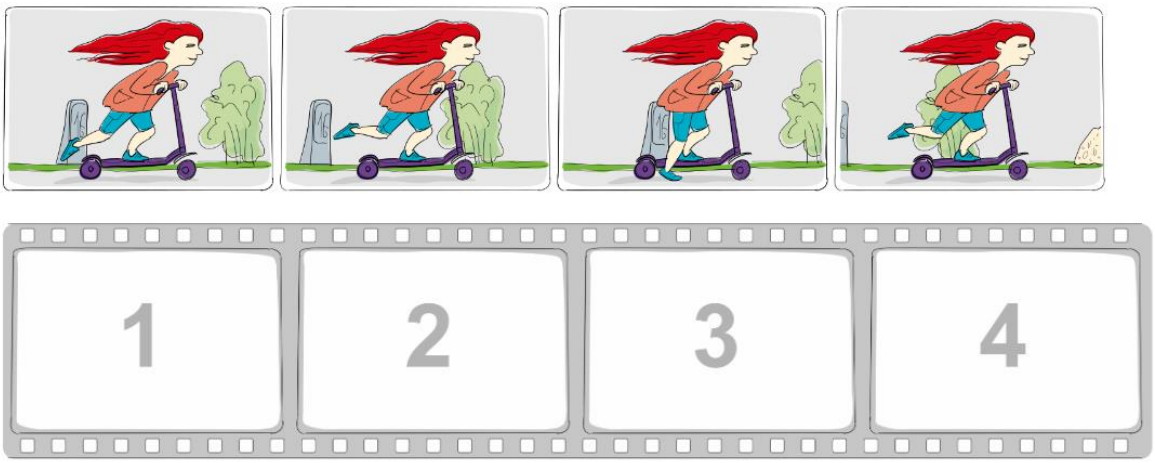
Kiekvieno uždavinio pradžioje nurodoma, kuriai amžiaus grupei jis skiriamas ir jo sudėtingumo lygis. Taip pat pateikiamas uždavinio atsakymas ir paaiškinimas, kaip uždavinys susijęs su informatika.

Nr.	Uždavinio pavadinimas	Uždavinio identifikatorius	Nykštukai	Mažyliai	Bičiuliai	Draugai	Jauniai	Kolegos
1	Išdėliok eilės tvarka	2020-LT-XX	6					
2	Bebro uodega	2020-LT-12	6					
3	Taškiniai blokai	2020-LT-10	6					
4	Pėdsakas	2020-UY-01	6					
5	Pliušinio meškiuko paieška	2020-IS-02	9	6				
6	Traukinių bėgiai	2020-PT-06	9	6				
7	Bitės skrenda	2020-SK-04	9	6				
8	Kalendorius	2020-LV-02	9	6				
9	Bebrų namų numeriai	2020-CH-09b	12	9	6			
10	Namų dažymas	2020-JP-02	12	9	6			
11	Parko grožybės	2020-TR-03	12	9	6			
12	Dubenėliai	2020-IE-09	12	9	6			
13	Saldainių dalybos	2020-KR-06		6				
14	Rąstų rūšiavimas	2020-PK-02		9	6			
15	Purkštuvai	2020-SK-03b		12	6			
16	Šokinėjanti kengūra	2020-LT-01		12	9	6		
17	Stebuklingas gėrimas	2020-PK-03		12	9	6		
18	Sujunk taškus	2020-MK-01		12	9			
19	Robotų stovėjimo aikštelė	2020-LT-06		12		6		
20	Apsilankymas parodoje	2020-CH-21			9	6		
21	Sunkiausia dėžė	2020-JP-04			9	6		
22	Teatro vaidinimas	2020-SK-01			9	6		
23	Apgadinta lentelė	2020-CH-03a			12	9	6	
24	Žvaigždės ir pusmėnuliai	2020-BE-03			12	9		
25	Bebras pilyje	2020-CH-07			12	9		
26	Maisto prekių parduotuvės	2020-AT-06			12		6	
27	Medžių sudoku	2020-CH-04b			12			
28	Ieškome lobio	2020-PK-05			12			
29	Bebrų miestelio laikrodis	2020-CH-18				9	6	6
30	Laimos dainelės	2020-CH-01c				9	6	

Nr.	Uždavinio pavadinimas	Uždavinio identifikatorius	Nykštukai	Mažyliai	Bičiuliai	Draugai	Jauniai	Kolegos
31	Miestai ir greitkeliai	2020-SK-02				9	6	
32	Šildymas	2020-PK-06				12	6	6
33	Bebrai ir kengūros	2020-LT-05				12	9	6
34	Bebrynės statybos	2020-UZ-02				12	9	6
35	Dėlionė	2020-CN-04				12	9	
36	Durininkas Jokūbas	2020-CY-02				12	9	
37	Buitinės technikos valdymas	2020-JP-01b				12	12	9
38	Kopėčios ir gyvatės	2020-IN-26					9	6
39	Naujienų plitimas	2020-MK-03					9	6
40	Medžių sudoku 4x4	2020-CH-04c					12	9
41	Kelionė traukiniu	2020-SI-01					12	9
42	Epidemija	2020-TW-02					12	9
43	Slaptažodžiai	2020-DE-05a					12	
44	Kompiuterių sparta	2020-NZ-03					12	
45	Sierpinskio trikampis	2020-HU-02						9
46	DNR seka	2020-ID-04						9
47	Skaitmeniniai medžiai	2020-AT-01						12
48	Hemingo lemingai	2020-RU-02						12
49	Vandens taksi	2020-CH-15						12
50	Pinigų maišeliai	2020-IR-03						12
51	Devyni stiklo kamuoliukai	2020-VN-04						12
52	Vėliavos	2016-CZ-04						12

1. Išdėliok eilės tvarka

Vilkdami pele išdėliokite paveikslėlius iš eilės, kaip vyksta veiksmas.



Paaiškinimas

Atsakymas:

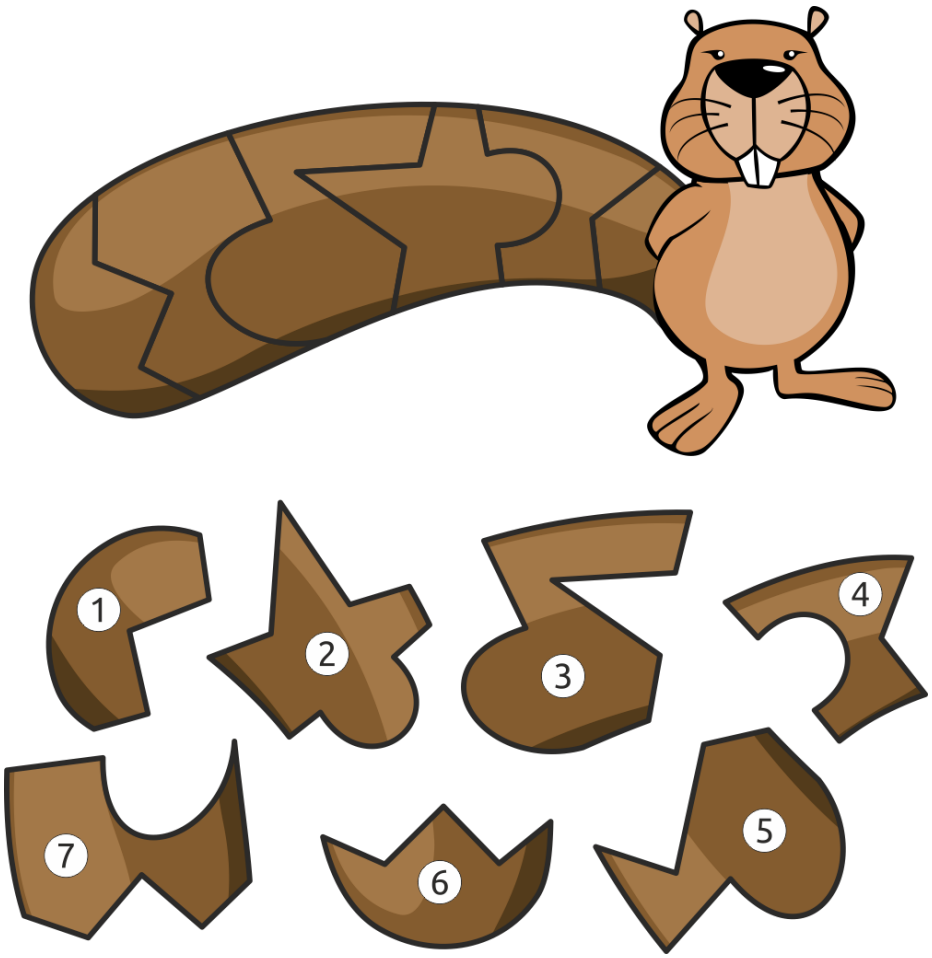
Tai informatika!

Programavime labai svarbu suprasti, kad komandos vykdomos iš eilės (yra komandų, keičiančių tvarką, bet tai aiškiai nurodoma). Komandos vykdomos taip, kaip jos pateiktos: įvykdoma pirmoji, tada vykdoma antroji ir t. t.

Komandų vykdymo eilės tvarkos supratimas – programavimo pradžiamokslis. Eiliškumą galima mokyti pateikiant statinius objektus, kai kas nors išdėstyta tam tikra seka, pavyzdžiui, įvairūs taisyklingi vėriniai, taip pat reikia pateikti ir dinaminius procesus, kaip šiame pavyzdyje, kai mergaitė juda – jos judėjimą atsekame pagal besikeičiantį foną.

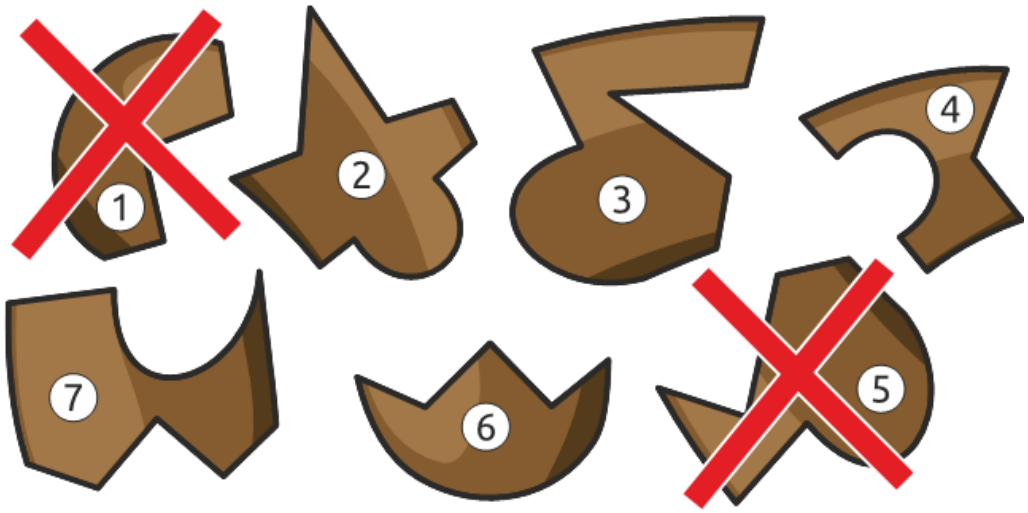
2. Bebro uodega

Kurios figūros nepriklauso uodegos raštui?



Paaiškinimas

Atsakymas: 1 ir 5.



Tai informatika!

Atlikdami šią užduotį, turime naudoti šablono atpažinimą - analizuoti ir atpažinti figūrų skirtumus.

Norėdami išspręsti užduotį, turite patikrinti kiekvieną figūrą ir įsitikinti, ar ji tilps į uodegą. Atitikimas randamas atsižvelgiant į kraštų dydį ir formą. Kiekviena padėtis turi būti patikrinta, norint įsitikinti, kad figūra nepriklauso bebro uodegai.

Ši užduotis – išsamios paieškos pavyzdys.

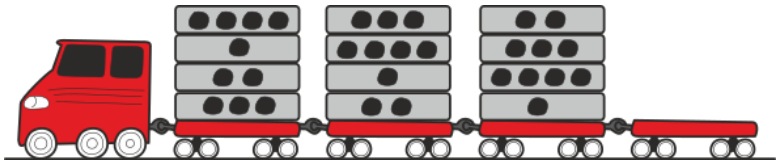
Struktūrų atpažinimas yra automatinis šablonų ir duomenų dėsningumų atpažinimas. Tam gali būti naudojama statistinė duomenų analizė, signalų apdorojimas, vaizdų analizė, informacijos gavimas, bioinformatika, duomenų glaudinimas, kompiuterinė grafika, mašininis mokymasis.

Kompiuterių moksle visiškas perrinkimas (kitaip: išsami paieška) taikoma sprendžiant labai bendro pobūdžio uždavinius. Išvardijami visi galimi sprendimo kandidatai ir tikrinama, ar kiekvienas kandidatas tenkina problemos teiginį. Kartais gali būti sudėtinga įsitikinti, kad sąraše nėra nė vieno kandidato.

Pagrindinis išsamios paieškos privalumas – garantuojama, kad bus rastas sprendimas, jei toks yra. Jei yra keli sprendimai, atlikus išsamią paiešką bus rasti visi. Išsamios paieškos gali užtrukti ilgai.

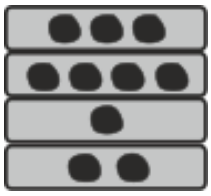
3. Taškiniai blokai

Traukinyje yra 4 vagonai. Į pirmus tris vagonus jau yra pakrauti blokai. Blokai pažymėti taškais ir išdėstyti pagal tam tikrą taisyklę.

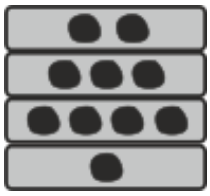


Kaip turėtų būti sukrauti blokai ketvirtame vagone, kad atitiktų taisyklę?

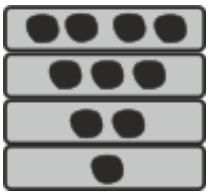
A.



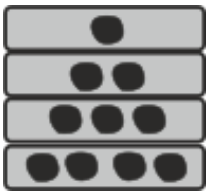
B.



C.



D.



Paaiškinimas

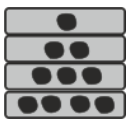
Atsakymas: D.

Norėdami surasti taisyklę, pagal kurią pakrauti blokai, galime pasirinkti bet kurį bloką, esantį pirmame vagone, ir stebėti, kur jis yra antrame ir trečiame vagone.

Taškų skaičius			
1 vagonas	2 vagonas	3 vagonas	4 vagonas
4	3	2	?
1	4	3	?
2	1	4	?
3	2	1	?

Pastebime, blokas, kuris turi vieną tašką, juda per vieną bloką žemyn. Todėl ketvirtojo vagono blokų bokštas turėtų būti išdėstytas taip:

Taškų skaičius			
1 vagonas	2 vagonas	3 vagonas	4 vagonas
4	3	2	1
1	4	3	2
2	1	4	3
3	2	1	4

Taigi ketvirtojo vagono blokai turėtų atrodyti taip: 

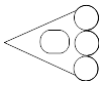
Tai informatika!

Informatikoje labai svarbu pastebėti panašius vaizdus ar šablonus. Šablonų atpažinimas yra glaudžiai susijęs su dirbtiniu intelektu ir mašininiu mokymusi. Rasti dėsningumus, nuspėti modelius ir elgesį nėra lengva, tačiau tai gali padėti kompiuteriams atlikti sudėtingas užduotis.

Šiuo atveju tinkamą sprendimą galima rasti ištyrus visus blokus ir atpažinus šabloną.

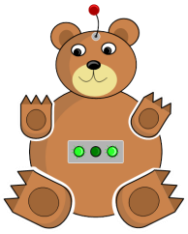
4. Pėdsakas

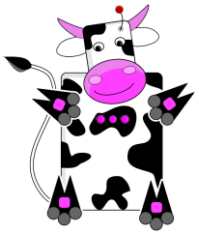
Parduotuvėje yra keturi gyvūnai-robotai.

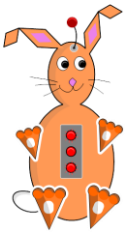
Naktį vienas iš jų slapčia vaikščiojo po parduotuvę. Ant grindų liko pėdsakas: 

Kieno šis pėdsakas?

A. 

B. 

C. 

D. 

Paaiškinimas

Atsakymas: D.

Vienintelio triušio pėdsakas yra sudarytas iš vieno didelio trikampio, jo viduje yra apskritimas, o krašte trys maži apskritimai.

Tai informatika!

Norint išspręsti šią užduotį, pirmiausia reikia identifikuoti kiekvieno iš siūlomų gyvūnų-robotų kojas ir, antra, patikrinti, kurios iš jų atitinka pėdsaką.

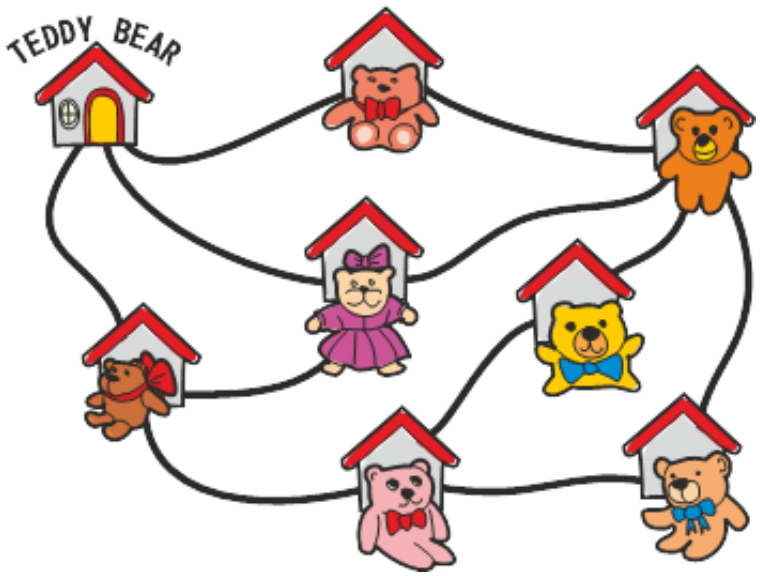
Tai darydami ignoruojame kitą nereikalingą informaciją, o tikriname tik pėdas. Šis procesas vadinamas abstrakcija, kuri dažnai naudojama kompiuterių moksle ir yra viena iš svarbių informatinio mąstymo įgūdžių. Abstrakcijai svarbu informacijos, reikalingos tam tikrai problemai išspręsti, parinkimas. Šiuo atveju būtina suvokti, kad svarbi tik pėdų forma.

Rašydami programas programuotojai nuolat abstrahuoja – sprendžia, į ką atsižvelgti. Tai taip pat yra dalis proceso, vadinamo modeliavimu, kuris padeda nuspręsti, kokią informaciją turi turėti kompiuteris, kad išspręstų užduotį.

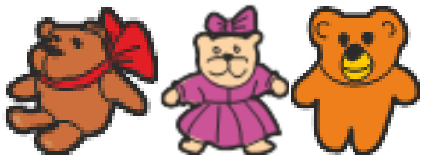
Supratimas, kad pėdsakas atitinka nurodyto gyvūno pėdą, atsiranda, kai lygindami vaizdus atpažįstame šablonus. Žmonės yra gana įgudę atpažinti, koks yra vaizdas (pavyzdžiui, ar nuotraukoje yra katė, ar ne), tačiau kompiuteriams tai gana sudėtinga užduotis. Buvo sukurti mašininio mokymosi metodai, kad kompiuteriai galėtų išspręsti tokias užduotis.

5. Pliušinio meškiuko paieška

Bebrų šeimyna eina ieškoti pliušinio meškiuko. Jie pradeda nuo savo namų (TEDDY BEAR), vaikšto tik esamais keliais ir turi grįžti atgal namo. Keliaudami jie daro pamatytų meškiukų nuotraukas. Štai taip atrodo kaimelio žemėlapis:



Bebrai pamatė 4 meškiukus, bet nuotraukų padarė tris - vieną meškiuką pamiršo. Štai nuotraukos:



Kurį meškiuką bebrai pamiršo?

A. 

B. 

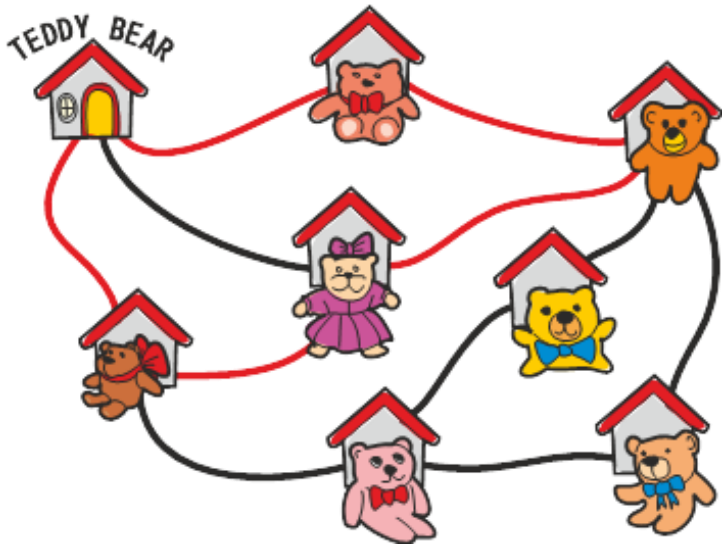
C. 

D. 

Paaiškinimas

Atsakymas: C.

Paveiksle raudona spalva pavaizduotas bebrų maršrutas. Tai yra vienintelis kelias, kuriuo einant aplankomi visi 4 meškiukai.



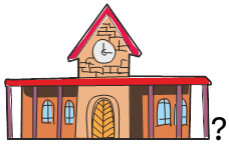
Tai informatika!

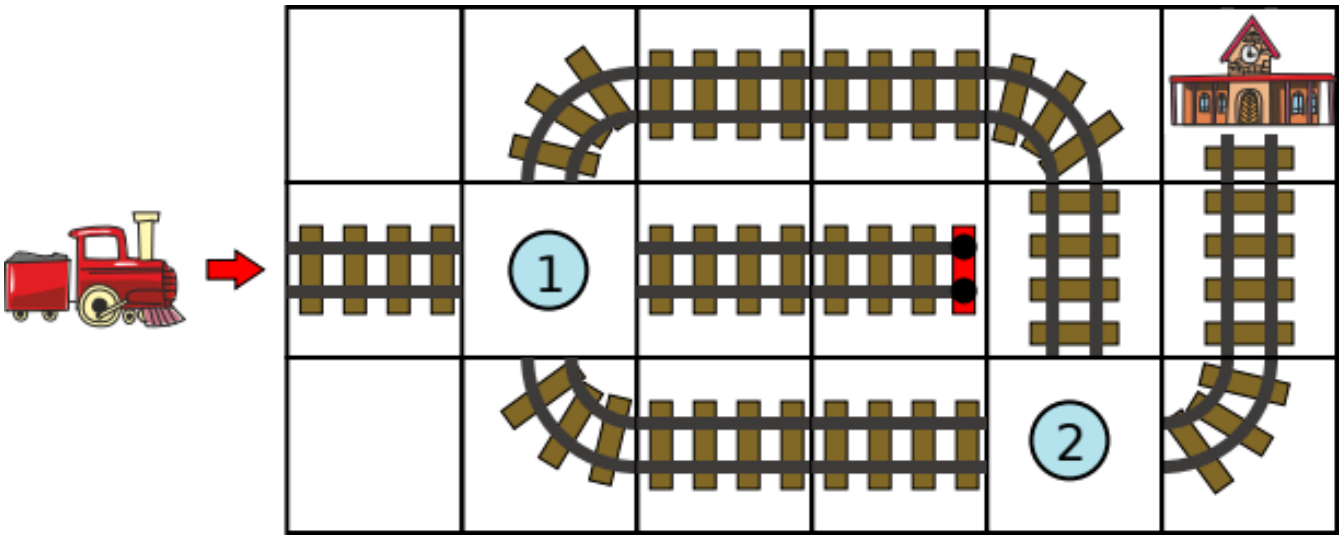
Kelio radimas yra bendra informatikos problema. Sprendžiant šį uždavinį reikia išbandyti visus galimus kelius ir rasti maršrutą, kuris tenkintų nurodytą sąlygą (aplankyti 4 meškiukus). Nors šis uždavinys atrodo paprastas, tačiau tampa sudėtingu esant bent kiek didesniam žemėlapiui.

Žemėlapis vaizduojamas grafu (viršūnių – namų ir briaunų – kelių tarp namų kombinacija), kuriame pasirenkame 4 viršūnes (bebrų namą ir tris namus, kurių nuotraukos duotos). Reikia rasti ketvirtą viršūnę, kuri nebūtų bebrų namas ir kuri būtų ant vieno iš kalių. Tai kombinatorikos uždavinys grafe.

Tai taip pat uždavinys apie klaidų paiešką. Beveik neįmanoma suprogramuoti uždavinio nepadarius klaidų. Klaidų paieška yra įprasta informatikos veikla. Klaidų ieškoma peržiūrint, testuojant sprendimą.

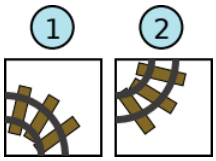
6. Traukinių bėgiai

Ar gali padėti traukiniui  pasiekti stotį  ?

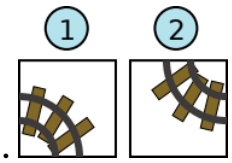


Kuri bėgių pora užpildys trūkstamas vietas taip, kad traukinys nuvyktų į stotį?

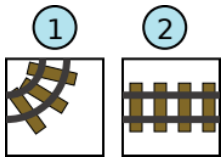
A.



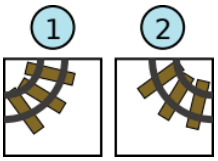
B



C.

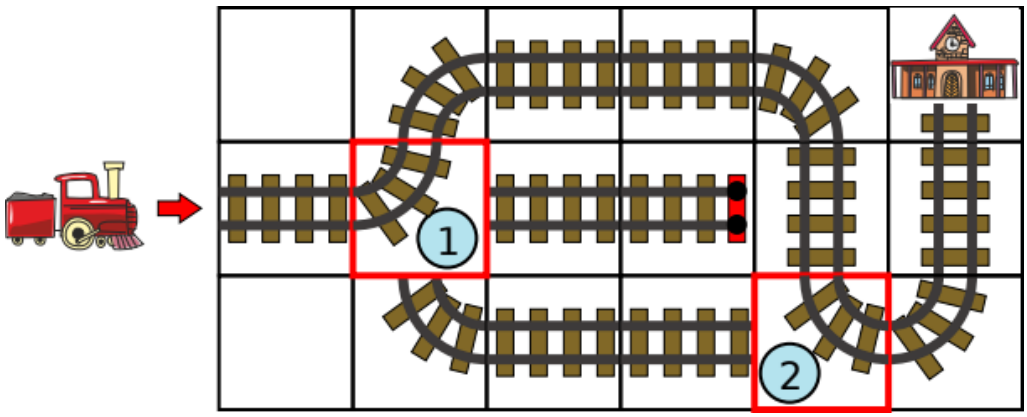


D.



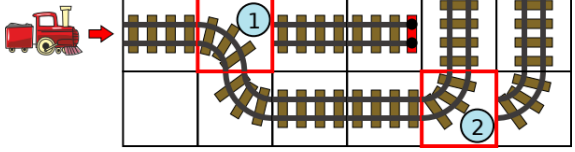
Paaiškinimas

D yra teisingas atsakymas, tai parodyta paveiksle:

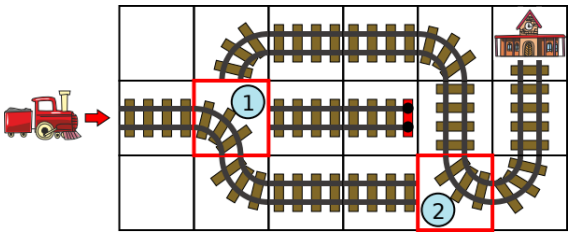


Štai kaip atrodytų kitų porų įterpimas:

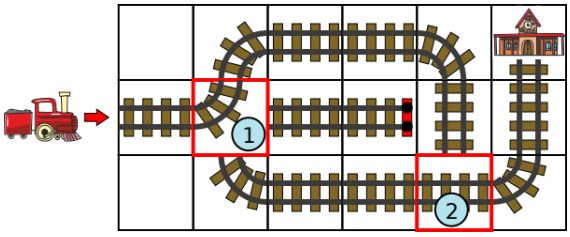
A)



B)



C)



Kaip matome, šiais atvejais traukinys nepasiektų stoties.

Tai informatika!

Mokiniams

Traukiniai važiuoja bėgiais. Bėgiai gali eiti tiesiai, suktis dešinėn ar kairėn. Kompiuterių programa yra panaši į traukinių bėgius. Programa nurodo kompiuteriui, ką daryti sprendžiant uždavinį, pavyzdžiui, sudedant du skaičius. Kompiuterių programa sudaryta iš daug komandų, panašiai kaip traukinių bėgiai sudaryti iš daug gabaliukų. Kartais, trūkstant kurios nors komandos, programa neveikia. Programuotojas turi surasti trūkstamą komandą, kad programa duotų teisingą rezultatą.

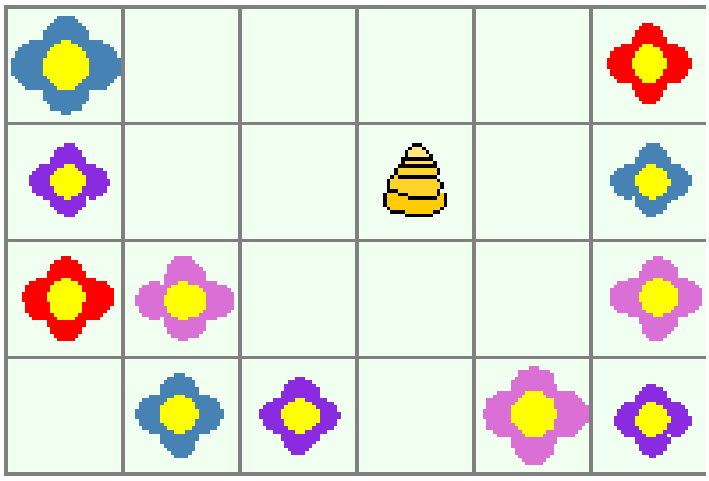
Mokytojams

Kurdami programas mokomės valdyti objektus ar veikėjus panašiai, kaip traukinių bėgiai valdo traukinių eismą. Kiekvienas bėgių gabaliukas gali būti laikomas komanda, kurią parenka programuotojas ir taip apibrėžia traukinio judėjimo kryptį. Komandų seka, kuri atliekama nustatyta tvarka ir kiekvieną kartą duoda tą patį rezultatą, vadinama programa.

Norėdami išspręsti duotą uždavinį ir rasti du trūkstamus bėgių gabaliukus, mokiniai turi numatyti, kaip traukinys judės prieš patekdamas į žiedą. Tai darydami, mokiniai analizuoja sistemą ir planuoja vykdomo objekto elgesį. Tai yra informatinio mąstymo komponentas.

7. Bitės skrenda

Bitės gali skristi iš vieno kvadratėlio į kitą tik horizontaliai ar vertikaliai. Kiekviena bitė per dieną gali nuskristi daugiausiai 3 kvadratėlius nuo avilio.



Kiek gėlių gali aplankyti bitės per dieną?

A) 5

B) 6

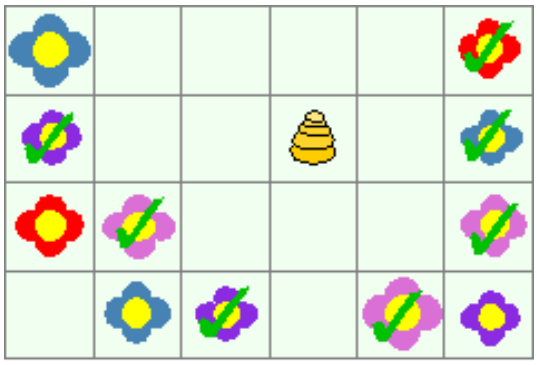
C) 7

D) 8

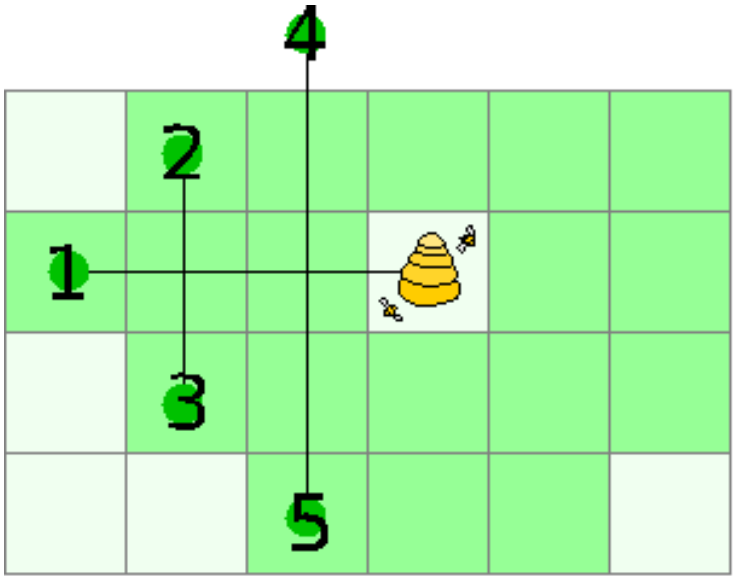
Paaiškinimas

Teisingas atsakymas: 7.

Paveiksle pažymėtos visos gėlės, kurias bitės gali pasiekti per 3 langelius horizontaliai ir vertikaliai.



Pirmiausia nustatykime visus tolimiausius kvadratėlius, kuriuos bitė gali pasiekti išskridusi iš avilio. Bitė gali skristi 3 kvadratėlius į kairę ir atsidurti taške 1. Ji gali skristi du kvadratėlius į kairę ir vieną aukštyn (2 taškas) arba vieną žemyn (3 taškas). Ji taip pat gali skristi vieną kvadratėlį į kairę ir du aukštyn (4 taškas, kuris išeina už lauko ribų) arba du žemyn (5 taškas). Tą patį metodą analogiškai pritaikome bitės skrydžiui iš avilio dešinėn, viršun ir žemyn, taip pat nepamirštame, kad bitė gali skristi ir trumpiau (vieną ar du kvadratėlius). Šitaip randame visus kvadratėlius, kurie yra nutolę ne daugiau kaip per tris kvadratėlius nuo avilio, paveiksle jie pažymėti ryškesne žalia spalva:

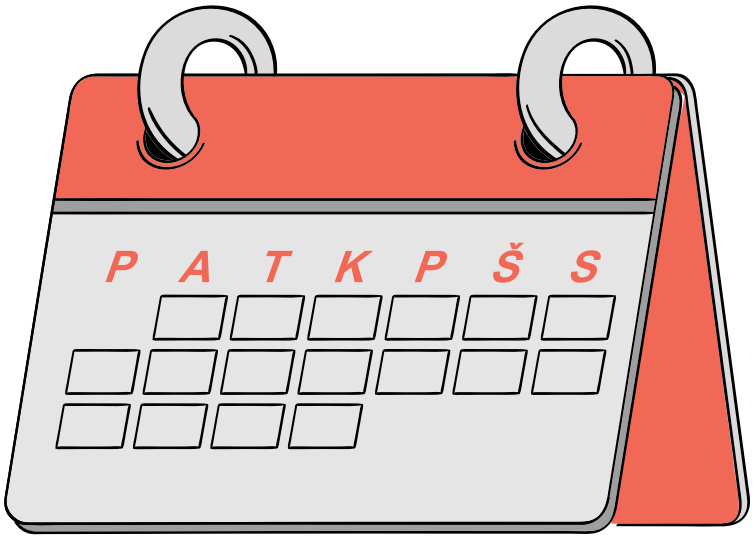


Tai informatika!

Bitės skrydis aprašomas algoritmu. Suprasti, kad algoritmai gali aprašyti kažkieno savybes (pavyzdžiui, kiek toli galima skristi, rasti visas vietas, kur galima skristi, ir pan.) yra svarbus gebėjimas informatikoje.
Pastebėkime, kad uždavinyje taikomas matavimas kvadratėliais nėra įprastas atstumas tarp dviejų taškų (vadinamo Euklido atstumu), kadangi bitės negali skristi įstrižai ar bet kaip kitaip, o tik horizontaliai ir vertikaliai.

8. Kalendorius

Diena, kuri prieš tris dienas buvo vakarykštė, buvo diena prieš sekmadienį.



Kuri diena bus rytoj?

A) Pirmadienis
B) Sekmadienis
C) Ketvirtadienis
D) Trečiadienis

Paaiškinimas

Atsakymas: ketvirtadienis.

Vakarykštė diena prieš tris dienas buvo prieš keturias dienas skaičiuojant nuo dabartinės dienos. Tai buvo šeštadienis (vakarykštė diena sekmadienio atžvilgiu).

Vadinasi, šiandien yra trečiadienis, o rytoj bus ketvirtadienis.

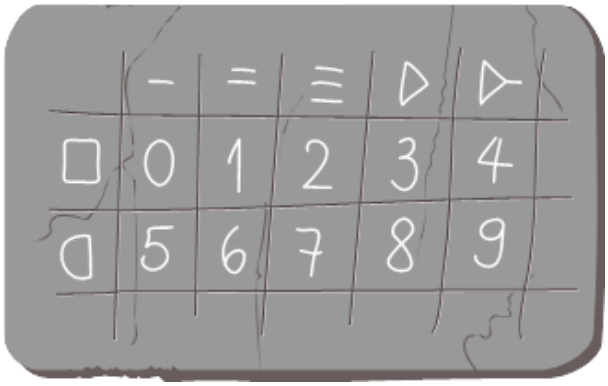
Tai informatika!

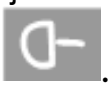
Šis uždavinys grindžiamas loginiu samprotavimu. Logika atlieka vieną pagrindinių vaidmenų informatikoje, jos principais remiamasi duomenų bazėse, programavimo kalbose, skaičiavimuose, dirbtiniame intelekte, projektuojant ir tikrinant aparatinę ir programinę įrangą ir pan. Logika yra neabejotinas pagrindas kuriant ateities kompiuterines sistemas, kalbas, metodus.

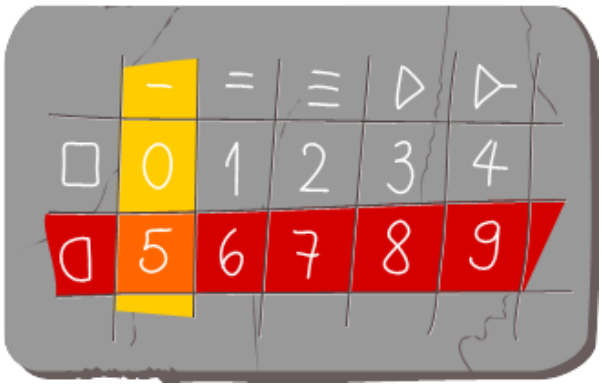
	Nykštukai	Mažyliai	Bičiuliai	Draugai	Jauniai	Kolegos
	12	9	6			

9. Bebrų namų numeriai

Bebrų namų numeracijai vietoj skaitmenų vartojami tik jiems suprantami simboliai. Jie sudaromi naudojantis šia lentele:



Pavyzdžiui, skaitmuo 5 užrašomas suradus jį lentelėje ir kombinuojant jo eilutės simbolį  su jo stulpelio simboliu  – gaunamas simbolis  .

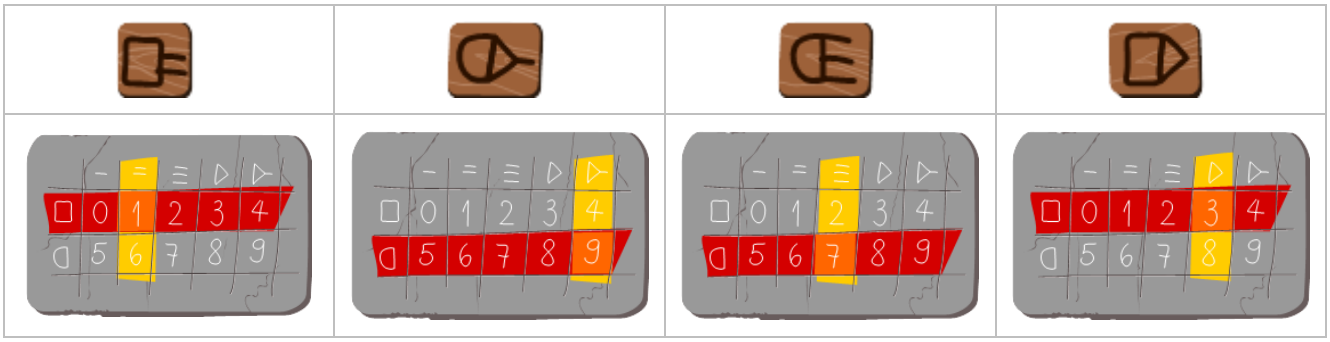


Pamatėme bebro namą su tokiu numeriu:

Koks šio bebro namo numeris?

Paaiškinimas

Bebrų namų numerių kiekvienas skaitmuo iškoduojamas suradus tinkamą eilutę (pagal pirmąją simbolio dalį) ir tinkamą stulpelį (pagal antrąją simbolio dalį) ir imant jų sankirtoje esantį skaitmenį.



Taigi bebro namo numeris yra 1973.

Tai informatika!

Tai kodavimo uždavinys. Kodavimu suprantamas procesas, kai kodas taikomas tiems patiems duomenims norint juos kitaip (ekvivalenčiai)pavaizduoti. Bebrų namų kodas šiuo atveju yra lentelė, kuri taikoma skaitmenims nuo 0 iki 9, norint juos pavaizduoti kitokiais simboliais.

Kodavimas yra labai svarbus informatikoje. Koduojami pranešimai norint paslėpti perduodamą turinį, kad ne visi jį perskaitytų – tai kriptografija.

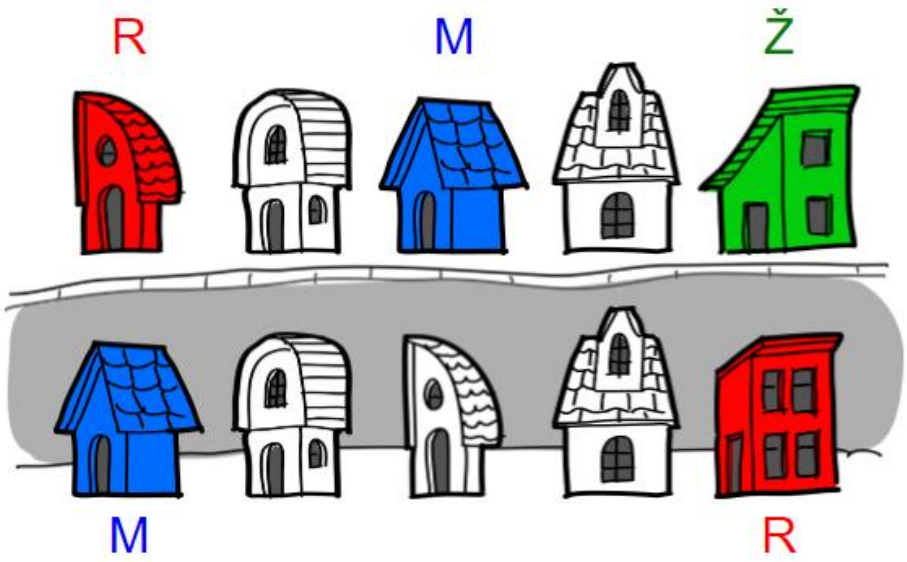
Koduojama norint sumažinti perduodamų duomenų dydį – tai duomenų glaudinimas.

10. Namų dažymas

Taro gatvės gyventojai dažo baltus namus laikydamiesi šių taisyklių:

1. Kiekvienas namas turi būti arba žalias, arba raudonas, arba mėlynas.
2. Du greta esantys namai neturi būti tos pačios spalvos.
3. Du namai, kurie stovi tiesiai vienas priešais kitą skersai gatvę, neturi būti tos pačios spalvos.

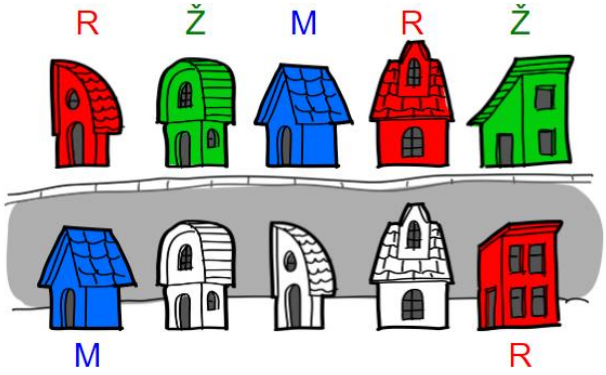
Kai kurie namai jau nudažyti ir jų spalvų negalima keisti.



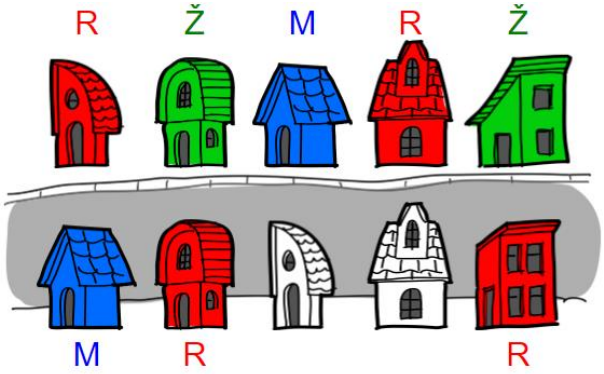
Spustelėkite pele baltą namą ir parinkite jo spalvą.

Paaiškinimas

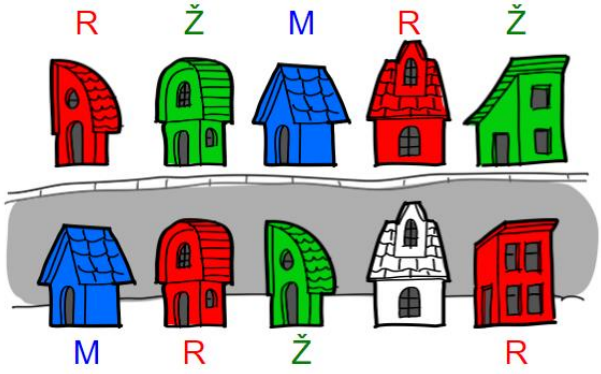
Pirmiausia dažome tuos namus, kurių kaimyniniai namai jau nudažyti ir tėra tik vienas pasirinkimas.



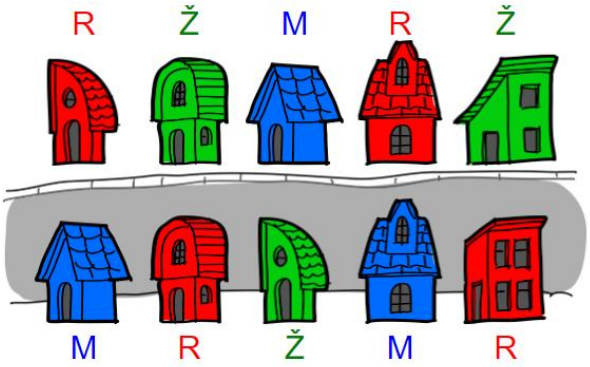
Toliau imamės antrojo namo kitoje gatvės pusėje: jis gali būti tik raudonas, nes greta jau yra mėlynas namas, o priešais – žalias.



Tolesnis namas gali būti tik žalias:

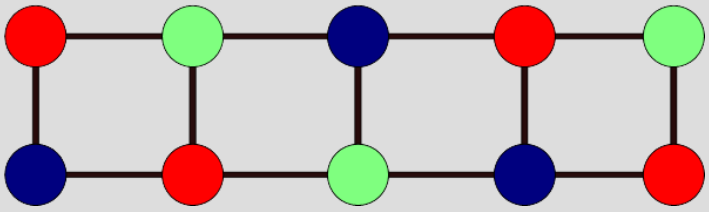


Pagaliau beliko paskutinis nenudažytas namas – jis turi būti mėlynas.



Tai informatika!

Kadangi kiekvieno namo spalva priklauso nuo kaimyninių ir priešais esančių namų, tai namus galime pavaizduoti skrituliukais ir linijomis pavaizduoti ryšius tarp jų. Gauname struktūrą, kuri informatikoje vadinama grafu. Skrituliukai, vaizduojantys namus grafe, vadinami viršūnėmis, o linijos – briaunomis.



Kiekviena linija šiame grafe nurodo ryšį, kad „namas yra greta“ arba „namas yra tiesiai prieš“. Iš tiesų abu ryšiai šiame uždavinyje yra to paties pobūdžio.

Turėdami grafą, galime performuluoti uždavinio klausimą: viršūnės turi būti dažomos taip, kad nei viena briauna nejungtų dviejų tos pačios spalvos viršūnių. Tokį klausimą galima kelti bet kokios formos grafui, tačiau ne visada įmanoma nuspalvinti turint tik tris spalvas.

Iš tiesų, seniai yra kilusi žemėlapio spalvinimo problema: kiek spalvų reikia žemėlapyje esančioms valstybėms nuspalvinti, kad bendrą sieną turinčios valstybės būtų nuspalvintos skirtingomis spalvomis. Prieš 200 metų buvo manoma, kad reikia penkių spalvų. 1976 metais matematikai Kenneth Appel ir Wolfgang Haken įrodė, kad užtenka keturių spalvų. Tai jie padarė pasitelkę kompiuterius ir atlikę daugybę skaičiavimų, kas nebūtų įmanoma be kompiuterio. Šis uždavinys pavadintas keturių spalvų problema, taikoma daugeliui kitokių uždavinių spręsti, pavyzdžiui, projektuojant lėktuvų skrydžių trajektorijas, kurios dalijamos išlaikant atstumus.

Informatikoje daug uždavinių gali būti modeliuojama grafais. Tai patogu, kadangi sukurta daug galingų algoritmų įvairiems grafų uždaviniams spręsti.

11. Parko grožybės

Tyrinėdami parką vaikai rado 4 gyvius ant 4 skirtingų medžių šakų:

- Vikšras tupi ant šakos, turinčios daugiausia lapų lyginant su kitomis šakomis.
- Drugelis sėdi ant šakos, turinčios daugiau lapų už šaką, ant kurios sėdi sraigė.
- Boružė ropoja šaka, kurios žievė yra šviesesnė nei bet kurios kitos šakos.

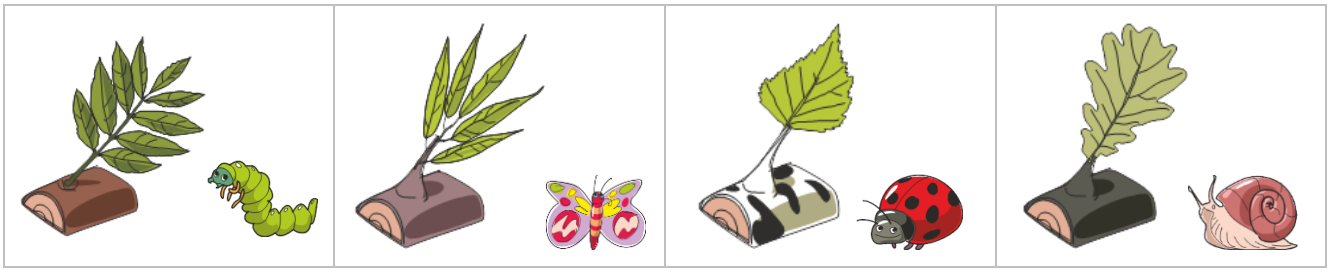
Kurie gyviai ant kurių šakų turi būti?

Paaiškinimas

Pirmoji šaka turi daugiausia lapų (9), tad ant jos turi tupėti vikšras.

Trečioji šaka yra šviesiausia (beržas), tad ji skirta boružei.

Drugelis ir sraigė turi būti ant likusių dviejų šakų. Žinome, kad drugelis yra ant šakos, turinčios daugiau lapų už šaką, ant kurios yra sraigė. Taigi drugelis yra ant antros šakos (5 lapai), o sraigė – ant paskutinės (1 lapas).



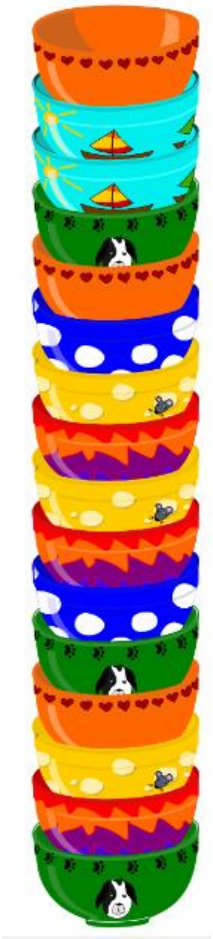
Tai informatika!

Tai – logikos uždavinys. Duotas teiginių rinkinys ir prašoma apibrėžti gyvių išsidėstymą taip, kad visi teiginiai būtų teisingi.

Logika yra teiginių nagrinėjimo mokslas, leidžiantis nustatyti, ar teiginiai yra visada, niekada ar kartais teisingi.

12. Dubenėliai

Trys broliukai nori valgyti pusryčius iš vienodų dubenėlių. Dubenėliai sudėti vienas į kitą kaip parodyta paveiksle. Dubenėlius galima imti tik nuo viršaus.



Kiek mažiausiai dubenėlių reikia nuimti, kol broliukai gaus tris vienodus?

A. 16

B. 15

C. 14

D. 13

Paaiškinimas

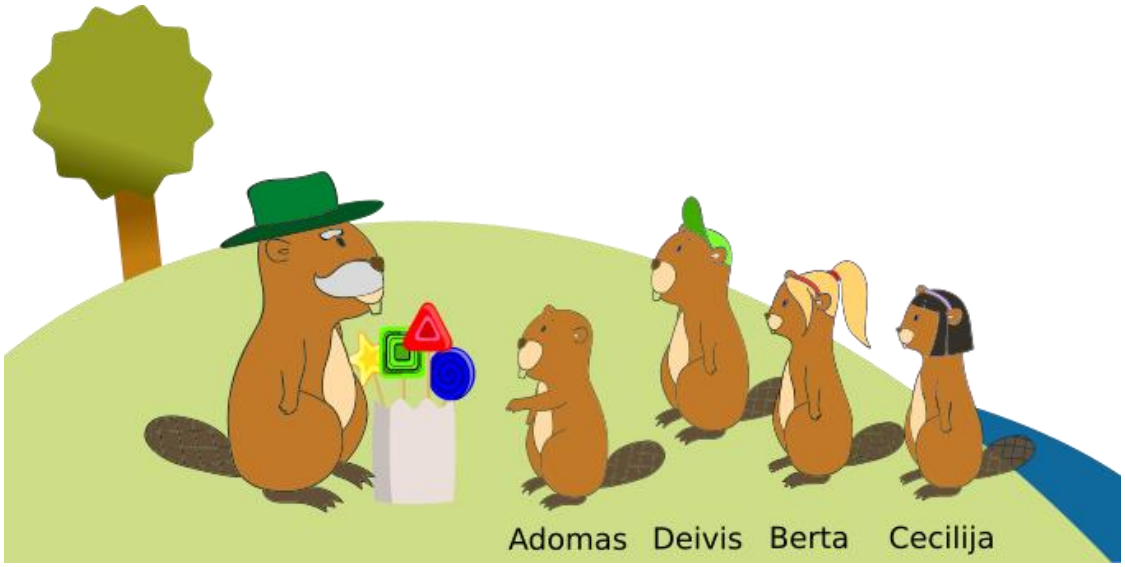
Atsakymas: D.

Bent 13 dubenėlių reikia nuimti, kol gausime tris vienodus (tai bus trys oranžiniai dubenėliai).

Tai informatika!

Dėklas (stekas) yra dažnai naudojama duomenų struktūra. Yra taisyklės, kaip dėti daiktus į dėklą ir kaip juos pašalinti iš dėklo. Šiame uždavinyje mums reikia tik paimti daiktus iš dėklo. Taisyklė tokia, kad vienu metu galima paimti tik vieną daiktą nuo dėklo viršaus. Jei norima gauti dešimtąjį dėklo dubenėlį, vadinasi, turime nuimti dešimt viršutinių dubenėlių. Taip pat svarbu, kur sudėti nereikalingus devynis dubenėlius – tai taip pat informatikos uždavinys. Jei mes turime antrą dėklą ir dėklai gali būti bet kokio aukščio, tai dubenėlių perkėlimo procesą galėtume suprogramuoti – informatikoje viskam stengiamasi parašyti programas. Taip yra todėl, kad dviem dėklais modeliuojama Tiuringo mašina – labiausiai nagrinėjamas teorinis kompiuteris. Taigi ši paprasta krūvelė – dėklas iš tiesų yra labai galinga!

13. Saldainių dalybos



Tėtis bebras nori padovanoti savo keturiems vaikams skirtingų formų ir spalvų saldainius.

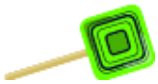
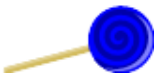
Adomas sako: „Aš nenoriu raudonų saldainių“.

Deivis sako: „Aš noriu žvaigždės formos saldainio“.

Berta sako: „Noriu, kad mano saldainis būtų kvadrato arba trikampio formos“.

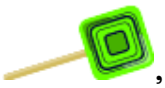
Cecilija sako: „Aš noriu raudonų saldainių“.

Kaip tėtis turi išdalinti saldainius, kad kiekvienas vaikas gautų norimą saldainį?

			
			
Adomas	Berta	Cecilija	Deivis

Paaiškinimas

Teisingas atsakymas:

Adomas -  , Berta-  , Cecilija -  , Deivis -  .

Kadangi Cecilija nori raudono saldainio, tėtis turėtų padovanoti jai trikampį saldainį. Berta nori kvadrato ar trikampio formos saldainio. Bet trikampis saldainis jau atiteko Cecilijai, todėl Berta turėtų gauti kvadratinį saldainį.

Liko mėlyno apskritimo ir geltonos žvaigždės formos saldainiai. Adomas teigė, kad bet kokie saldainiai yra puikūs, išskyrus raudonus, bet Deivis nori žvaigždės formos saldainio. Tėtis turėtų atiduoti žvaigždės formos saldainį Deiviui, tad apskritimo formos saldainis teks Adomui.

Tai informatika!

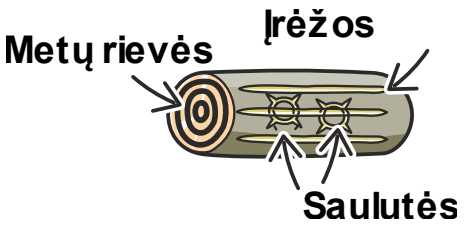
Ši užduotis yra pavyzdys to, ką kompiuterių mokslo mokslininkai vadina „ribojimų patenkinimo problema“. Spręsdami ribojimų patenkinimo problemą nagrinėjame daug teiginių, pavyzdžiui, keturių vaikų pareiškimus atliekant šią užduotį.

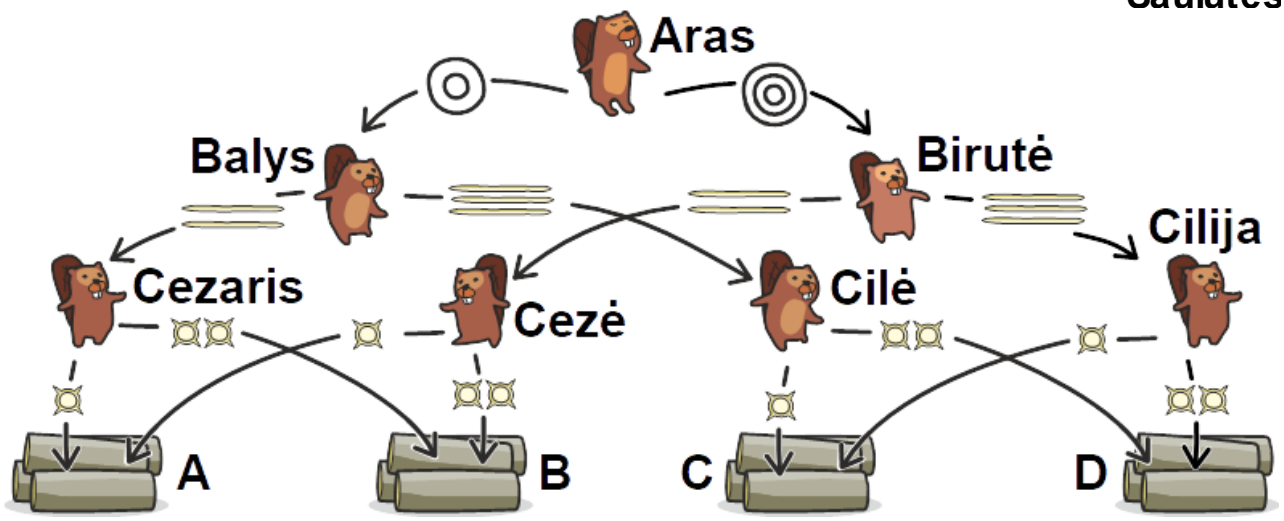
Kiekvienas teiginys gali turėti vieną ar daugiau „kintamųjų“, kuriuos galime laisvai pasirinkti. Pavyzdžiui, šios užduoties kintamieji yra saldainiai, kuriuos reikia duoti kiekvienam vaikui. Ribojimų patenkinimo problemos sprendimo tikslas yra pasirinkti, koks turėtų būti kiekvienas kintamasis, kad būtų patenkinta kuo daugiau, jei ne visi teiginiai. Paprastai yra ribotas kintamųjų pasirinkimo rinkinys ir teiginiai gali prieštarauti vienas kitam, jei pasirinkimai nėra padaryti kruopščiai.

Dėl konfliktų gali būti sunku išspręsti ribojimų patenkinimo problemas. Vykdant šią užduotį, Cecilijos teiginys (jis nori raudono saldainio) konfliktuoja su Bertos teiginiu (ji nori turėti trikampį saldainį), nes yra tik vienas raudonas trikampio formos saldainis. Laimei, yra būdas išspręsti konfliktą, kad abi būtų patenkintos.

14. Rąstų rūšiavimas

Bebrų kaimelyje yra rąstų saugykla, kurioje rąstai rūšiuojami pagal tris žymes (metų rievių, įrėžų ir saulučių skaičių) į keturias krūvas (A, B, C, D) kaip parodyta schemoje.





Šis rąstas  bus padėtas į D krūvą, nes yra trys metų rievės (Aras nukreips į dešinę), trys įrėžos (Birutė nukreips į dešinę) ir dvi saulutes (Cilija padės į krūvą D).

Į kurią krūvą bus padėtas šis  rąstas, remiantis jo savybėmis?

Paaiškinimas

Teisingas atsakymas: C.

Išnagrinėkime visus atvejus.

A krūva turi dviejų tipų rąstus:

- Pirma, tie, kurie turi 2 metų rieves, 2 įrėžas ir 1 saulutę;
- Antra, tie, kurie turi 3 metų rieves, 2 įrėžas ir 1 saulutę.

B krūva turi dviejų tipų rąstus:

- Pirma, tie, kurie turi 2 metų rieves, 2 įrėžas ir 2 saulutes;
- Antra, tie, kurie turi 3 metų rieves, 2 įrėžas ir 2 saulutes.

C krūva turi dviejų tipų rąstus:

- Pirma, tie, kurie turi 2 metų rieves, 3 įrėžas ir 1 saulutę;
- Antra, tie, kurie turi 3 metų rieves, 3 įrėžas ir 1 saulutę.

D krūva turi dviejų tipų rąstus:

- Pirma, tie, kurie turi 2 metų rieves, 3 įrėžas ir 2 saulutes;
- Antra, tie, kurie turi 3 metų rieves, 3 įrėžas ir 2 saulutes.

Kaip matome, tinka tik C atsakymas.

Tai informatika!

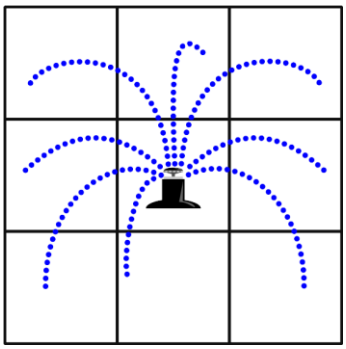
Šiame uždavinyje galime įžvelgti kelis svarbius informatikos konceptus. Pirmiausia, tai tarytum sprendimo medis, kas informatikoje daug kur taikoma. Tiesa, šiame uždavinyje sprendimo medis pateikiamas kiek kitaip, jo žemiausios šakos rodo ne į vieną objektą, kaip turėtų būty medyje, o į kelis, todėl šiuo atveju geriau būtų vadinti sprendimo diagrama. Galime įžvelgti ir kitą informatikos konceptą – funkciją su keliais kintamaisiais. O svarbiausia šiame uždavinyje yra atributų konceptas. Rąstas turi tris atributus (rieves, įrėžas ir saulutes), be to, kiekvienas atributas gali įgyti dvi reikšmes.

Sprendimo medžiai panaudojami daugelyje informatikos uždavinių, pavyzdžiui, siunčiant duomenų paketus kompiuterių tinklais.

15. Purkštuvai

Bebras Bronius sodina gėlių darželius kvadratėliais suskirstytame lauke (žr. paveikslą).

Kiekvienas gėlių darželis turi gauti vandens, todėl Bronius nusprendžia tuščiuose laukeliuose įrengti keletą laistymo purkštuvų. Purkštuvas gali pasiekti aštuonis gretimus gėlių darželius, kaip parodyta paveikslėlyje:

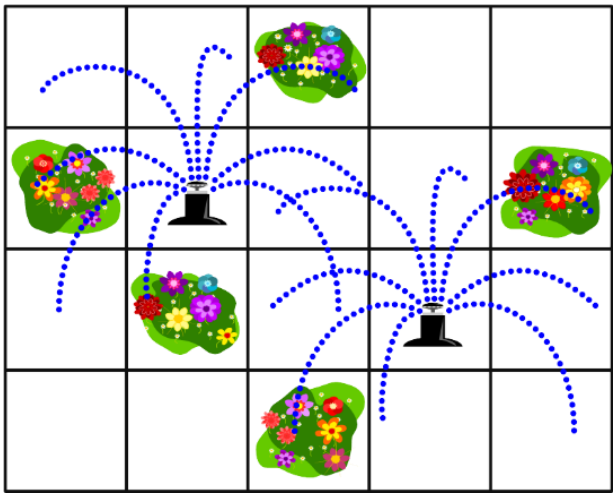


Kiek mažiausiai purkštuvų reikia pastatyti, kad būtų palaistyti visi pasodinti gėlių darželiai?

Paaiškinimas

Teisingas atsakymas – reikia 2 purkštuvų.

Vieno purkštuvo neužtenka. Vienas purkštuvas aprėpia 3x3 langelių plotą. Jei pradedame laistyti kairėje esančius darželius, tai iš karto pastebime, kad dešiniausias darželis per daug nutolęs, kad jį pasiektų purkštuvas, kuris laistys kairiuosius darželius. Taigi reikės dviejų purkštuvų. Klaidų ir bandymų metodu randame sprendimą.



Tai informatika!

Informatikoje dažnai tenka ieškoti geriausio problemos sprendimo. Tai yra, reikia sugalvoti, kaip palyginti du sprendimus, rasti kriterijus ir išrinkti geriausią. Šiame uždavinyje prašoma ne tik palaistyti visus gėlių darželius, bet ir įrengti kuo mažiau purkštuvų (kad būtų taupomas vanduo).

Paprasčiausias sprendimas būtų įrengti purkštuvą prie kiekvieno darželio, tačiau tai neekonomiška ir neekologiška. Galbūt kai kurie darželiai būtų perlaistomi (bet tai jau kitas kriterijus).

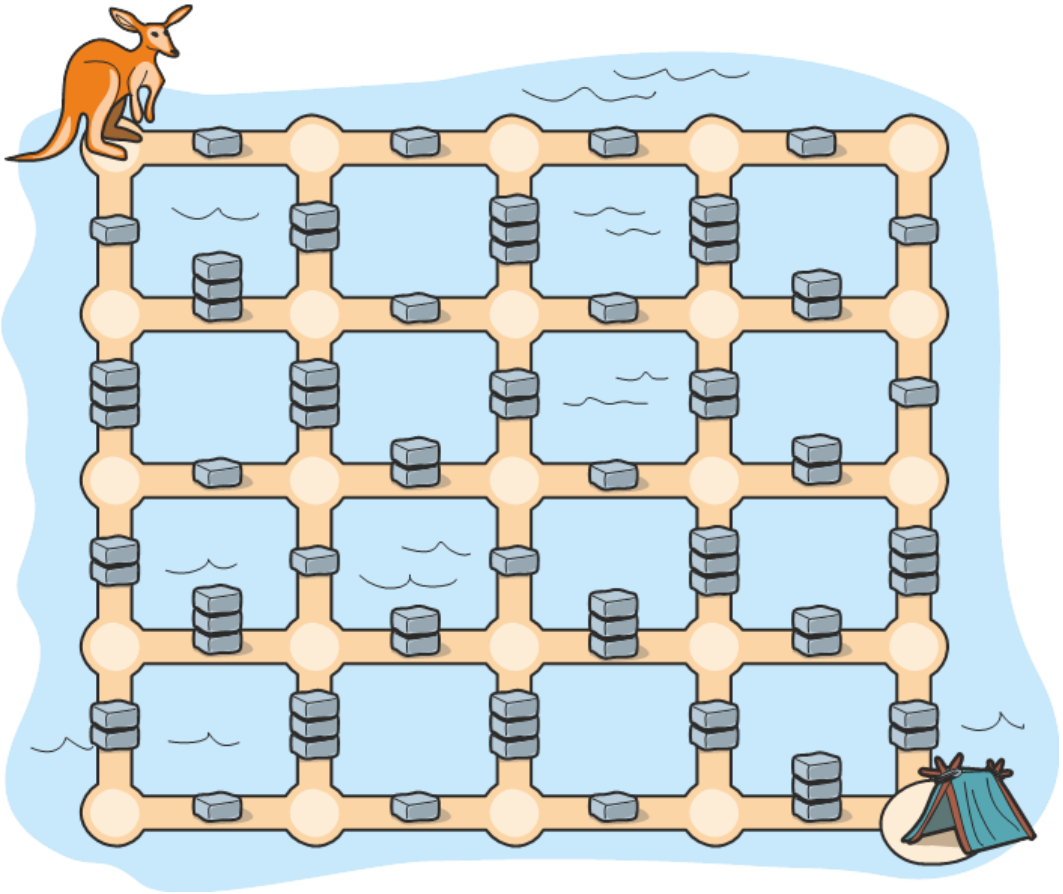
Šiame uždavinyje nesunku rasti sprendimą klaidų ir bandymų metodu. Galima išbandyti visus tuščius langelius, statant juose purkštuvus. Jie plotas būtų didesnis, daugiau būtų darželių, tai uždavinys būtų sunkus. Informatikoje yra metodų, kuriais sprendžiamos tokio pobūdžio optimizavimo problemos. Bendru atveju ne visada randamas optimaliausias sprendimas, dažnai pasitenkinama artimu optimaliam sprendimui.

Realiame gyvenime tokio pobūdžio uždaviniai sprendžiami išdėstant sveikatos centrus, ugniagesių tarnybas, vaistines ir pan., norint kuo geriau tenkinti gyventojų poreikius.

16. Šokinėjanti kengūra

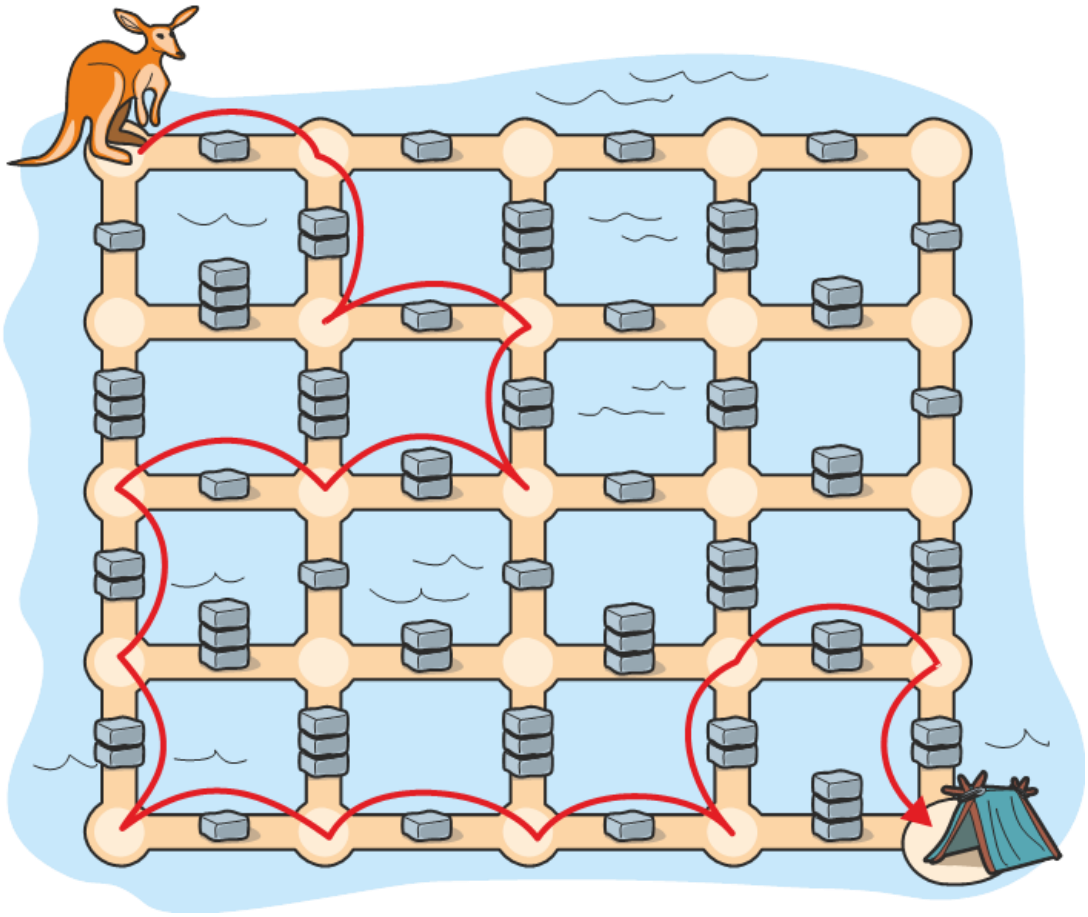
Kengūra nori grįžti namo kaip įmanoma greičiau. Ji gali keliauti šokinėdama tik tais takais, ant kurių padėta ne daugiau kaip dvi plytos. Kengūra gali šokinėti tik vertikaliai (aukštyn – žemyn) arba horizontaliai (kairėn – dešinėn).

Kiek mažiausiai šuolių padarys kengūra, grįždama namo trumpiausiu keliu?



Paaiškinimas

Atsakymas: 14.



Tai informatika!

Grįžimo metodas (angl. *backtracking*) – tai sistemingas būdas spręsti uždaviniams, kurių sprendinys yra kintamųjų p_1, p_2, ..., p_n reikšmių rinkinys, tenkinantis kokius nors reikalavimus.

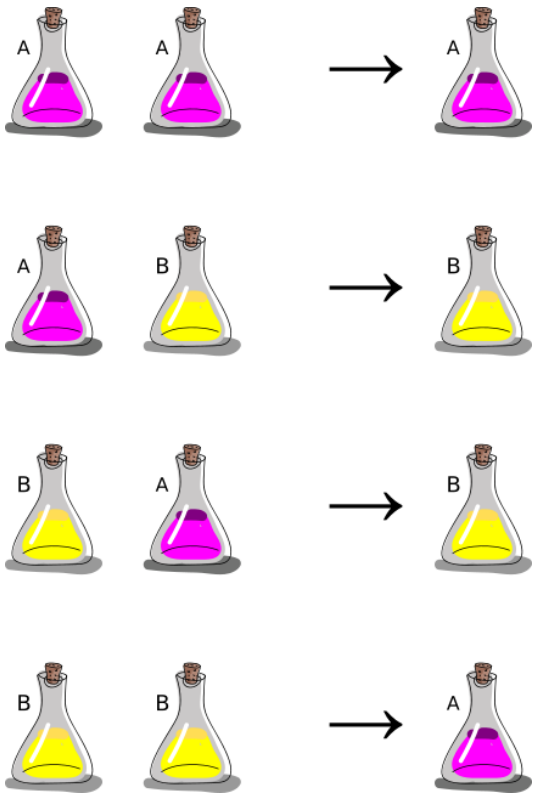
Pagrindinė grįžimo metodo idėja tokia: paeiliui renkamos visų galimų kintamųjų reikšmės ir tikrinama, ar tenkinami reikalavimai, o radus sprendinį arba situaciją, kai reikalavimai netenkinami (ant tako padėtos 3 plytos), grįžtama per vieną žingsnį atgal ir parenkama nauja atitinkamo kintamojo reikšmė.

Šis metodas taikomas sprendžiant sudoku, dėliones ar kombinatorikoje (optimizavimo uždaviniai).

Kai kuriais atvejais, kaip ir šiame uždavinyje, efektyviau pradėti perrinkimą nuo rezultato (pabaigos), kai iš kengūros namų pradedamas maršrutas. Einant tokiu keliu, padaroma mažiau grįžimų ir paprasčiau surandamas sprendimas. Tačiau bendru atveju, visapusiškai neišnagrinėjus užduoties, neįmanoma atsakyti, ar efektyviau pradėti nuo pradžios, ar nuo galo.

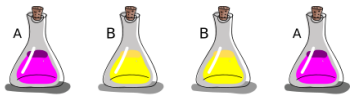
17. Stebuklingas gėrimas

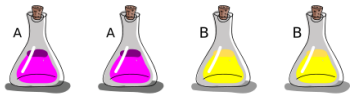
Burtininkas turi dviejų A ir B tipų stebuklingo gėrimo. Jei burtininkas gėrimus vieną po kito supila į vieną indą ir pasako burtažodį, gauto gėrimo spalva kiekvienu atveju atrodo taip, kaip pavaizduota paveikslėlyje:

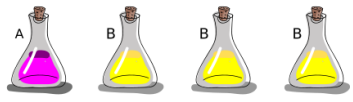


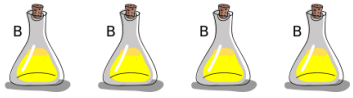
Burtininkas sumaišė 4 butelius stebuklingo gėrimo ir gavo B tipo gėrimą  .

Kokių tipų butelius ir kokia tvarka jis maišė?

A. 

B. 

C. 

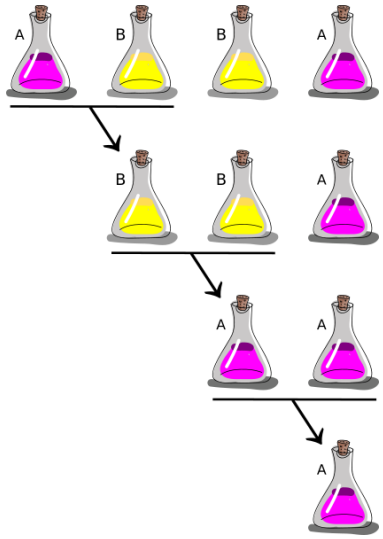
D. 

Paaiškinimas

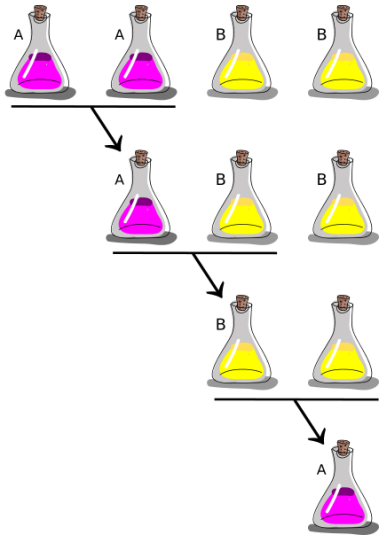
Atsakymas: C.

Gaminant stebuklingą gėrimą kiekvienu atveju gaunami tokie rezultatai:

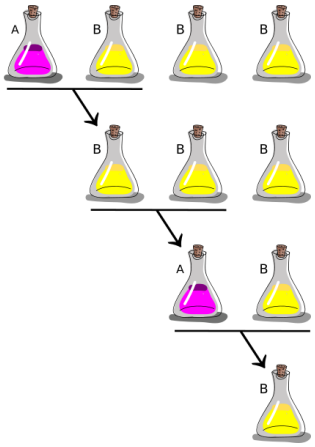
A.



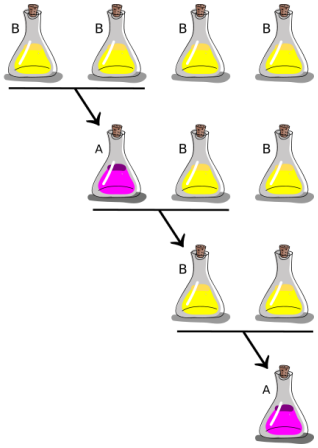
B.



C.



D.



Tai informatika!

Mokiniams

Kompiuteriai – tai elektroniniai įrenginiai, kuriuose informaciją tvarko elektra. Jei elektros srovė teka, sakome „Įjungta“, o jei ne – „Išjungta“. Informatikoje šios dvi būsenos dažniausiai vaizduojamos skaitmenimis 0 („Išjungta) ir 1 (Įjungta). Viena būsena atitinka vieną informacijos porciją, kuri vadinama bitu. Bitais pavaizduota informacija vadinama dvejetaine. Su bitais atliekami įvairūs veiksmai (operacijos), panašiai, kaip burtininkas atlieka įvairias kombinacijas su stebuklingo gėrimo buteliais. Viena tokių operacijų informatikoje vadinama logine XOR operacija. Šios operacijos rezultatai atitinka burtininko veiksmus ir atrodo taip:

0 XOR 0 = 0

0 XOR 1 = 1

1 XOR 0 = 1

1 XOR 1 = 0

Pora pavyzdžių. Laiptinės apačioje ir viršuje yra du jungikliai, kurie įjungia arba išjungią tą pačią lemputę. Kai abu jungikliai išjungti arba įjungti, lemputė šviečia, o kai vienas įjungtas, o kitas išjungtas, laiptinėje tamsu.

Kitas pavyzdys. Mama vaiko prašo pasirinkti tik vieną vaisių ir įsidėti į priešpiečių dėžutę. Ant stalo yra apelsinas ir obuolys. Jei vaikas nieko neišsirinks, dėžutė bus tuščia, o jei jis panorės įsidėti abu vaisius, tai jie į dėžutę netilps. Lieka teisingi tik du variantai: arba dėžutėje atsiduria obuolys, arba apelsinas.

Kompiuteriuose šią operaciją atlieka XOR loginis elementas. Šis elementas turi du įėjimus, elemento išėjime gauname būseną 1 („Įjungta“) tik tuomet, kai vieno įėjimo būsena yra 0 („Išjunta“), o kito – 1 („Įjungta“).

Mokytojams

Griežtosios disjunkcijos operacija (angl. *exclusive disjunction*, *exclusive OR*, santrumpa XOR), dar vadinama išskirtinio ARBA operacija, gali būti naudojama rasti ir taisyti klaidas.

Knygos „Informatika be kompiuterio“ 4 veikla „Kortelių keitimo magija. Klaidų radimas ir taisymas“ paremta XOR operacija.

XOR operacija naudojama ir kuriant NIM žaidimo strategiją. NIM žaidime dalyviai iš eilės ima daiktus, išdėliotus į kelias krūveles. Iš kiekvienos krūvelės gali būti paimtas bet koks daiktų skaičius (bet ne mažiau kaip vienas). Laimi žaidėjas, kuriam atitenka paskutinis daiktas. Laimi strategija, kuri kiekvieną ėjimą baigia krūvelių dydžio XOR suma, lygia nuliui. Informatikoje XOR operacija naudojama, kai reikia nurodyti, ar du bitai yra tokie patys, ar skirtingi. Loginėse grandinėse bet kurią loginę operaciją galima sudaryti tik iš XOR elementų.

XOR naudojama ir kriptografijoje.

„Informatika be kompiuterio“:
https://classic.csunplugged.org/wp-content/uploads/2015/09/KNYGA-Informatika-be-kompiuterio-2015-09-03.pdf

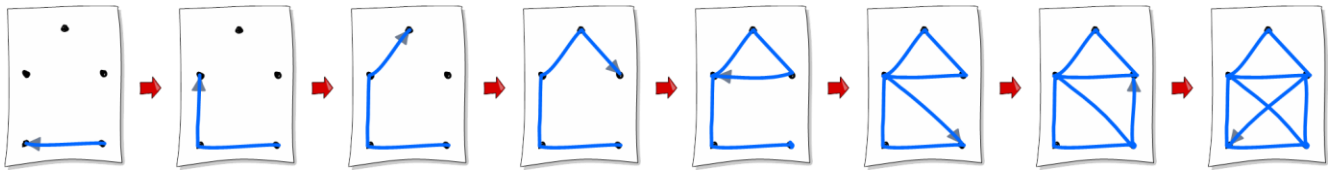
Informatinio mąstymo aspektai

Nagrinėjant šią užduotį galima supažindinti su procedūros samprata, išbandyti jos formalų aprašą, taip pat pritaikyti ją šiai užduočiai išspręsti.

18. Sujunk taškus

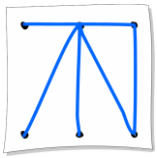
Išbandykime štai tokį piešimo būdą: piešinį reikia nupiešti neatitraukus pieštuko nuo lapo; piešinį sudaro atkarpos, brėžiamos nuo vieno taško iki kito, o tą pačią atkarpą galima brėžti ne daugiau kaip vieną kartą.

Pavyzdžiui, šitaip vientisai brėždami galime nupiešti namą, jei atkarpas brėšime tokia tvarka:

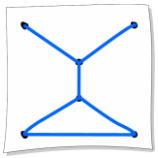


Kurią iš toliau pavaizduotų figūrų taip pat galima nupiešti tokiu būdu?

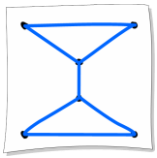
A.



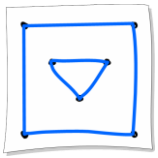
B.



C.

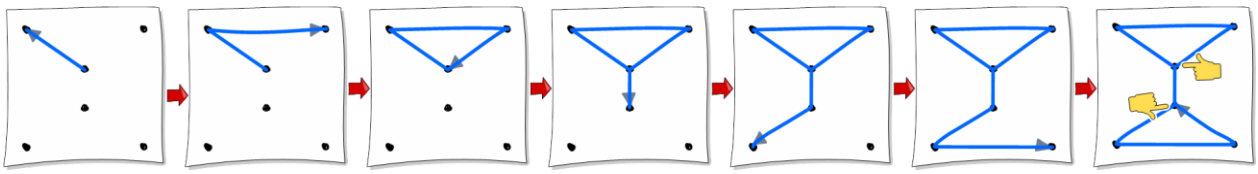


D.



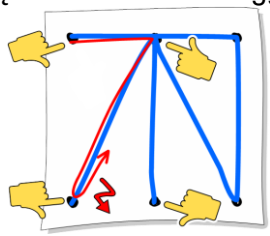
Paaiškinimas

Atsakymas: C. Viena iš galimų sekų, kaip brėžti atkarpas, galėtų būti tokia:



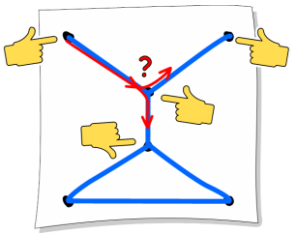
Atkreipkite dėmesį, kad atsakymo C piešinyje yra du taškai, į kuriuos sueina po nelyginį skaičių atkarpų. Tie taškai parodyti paskutiniame sekos paveiksliuke. Į visus kitus taškus sueina lyginis skaičius atkarpų.

Atsakymo A piešinyje yra keturi taškai, į kuriuos sueina nelyginis skaičius atkarpų.



Vis dėlto, mūsų norimu būdu piešdami piešinį privalome viename taške pradėti piešti, o kitame baigti. Jei norime nupiešti piešinį čia aprašomu būdu, turi būti ne daugiau kaip du taškai, į kuriuos sueina nelyginis skaičius atkarpų. Jei tokių taškų būtų daugiau, piešinį galėtume nupiešti tik brėždami tą pačią atkarpą keletą kartų, o to pagal mūsų sąlygą daryti negalime.

Situacija atsakymo B piešinyje labai panaši – brėždami piešinėlyje pažymėtas atkarpas galime brėžti arba žemyn, arba viršun dešinėn, bet vis tiek turėsime grįžti jau nubrėžta atkarpa, kad galėtume nubrėžti trečiąją.



Atsakymo D piešinys sudarytas iš trikampio, nubrėžto kvadrato viduje, tačiau nė vienas vidinio trikampio taškas nėra sujungtas su kuriuo nors išorinio kvadrato tašku, tad tokio piešinio neįmanoma nupiešti nepakėlus pieštuko.

Tai informatika!

Šie piešiniai sudaryti iš taškų ir atkarpų, jungiančių tuos taškus. Kompiuterijoje tokiu būdu yra vaizduojami objektai ir sąryšiai tarp objektų: taškai reiškia objektus, o atkarpos – sąryšius tarp objektų. Toks atvaizdas vadinamas grafu. Grafas yra sudarytas iš viršūnių (paprastai žymimų taškais arba apskritimais) ir briaunų (vaizduojamų atkarpomis arba kreivėmis, kartais turinčiomis kryptį nurodančią rodyklę). Briaunos sujungia viršūnes. Vientisa grafo briaunų seka vadinama keliu.

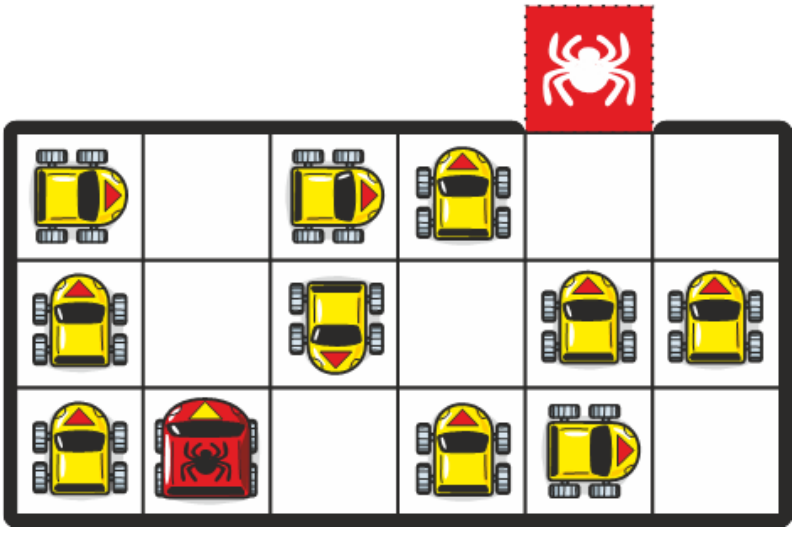
Čia mūsų piešti grafai yra ypatingi: grafas turi būti jungus (iš bet kurios grafo viršūnės briaunomis galima pasiekti bet kurią kitą to paties grafo viršūnę) ir tik dvi arba nė viena viršūnė gali turėti nelyginį laipsnį (taip vadinamas iš viršūnės išeinančių atkarpų skaičius). Jei grafas atitinka šias dvi sąlygas, tokį grafą visuomet bus galima nubrėžti vientisu judesiu ir nebrėžiant tos pačios atkarpos daugiau nei kartą. Toks piešinys vadinamas Oilerio keliu – pagal Leonardą Oilerį (1707 – 1783), pirmąjį mokslininką, nagrinėjusį tokio tipo uždavinį. Oileris šią idėją sugalvojo mėgindamas išspręsti septynių Karaliaučiaus tiltų uždavinį. Šiais laikais Oilerio kelią galima apskaičiuoti taikant Fleury arba Hierholzerio algoritmus.

19. Robotų stovėjimo aikštelė

Automobiliai-robotai aptvertoje teritorijoje gali judėti tik pagal nustatytas taisykles. Automobiliai vienu veiksmu gali atlikti vieną iš manevrų:

- pavažiuoti vieną langelį į priekį;
- pavažiuoti vieną langelį atgal;
- pasisukti 90 laipsnių į kairę savo langelyje;
- pasisukti 90 laipsnių į dešinę savo langelyje.

Be to, išvažiuoti iš teritorijos į voro ženklu pažymėtą langelį gali tik automobilis „Voras“.



Kiek mažiausiai veiksmų turės atlikti automobiliai-robotai, kol automobilis „Voras“ atsidurs voro ženklu pažymėtame langelyje?

A. 9

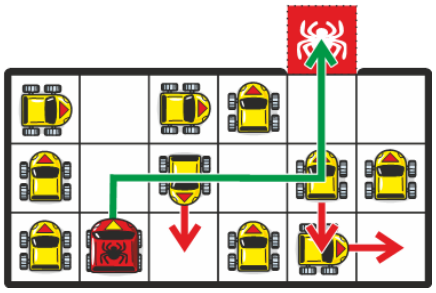
B. 11

C. 13

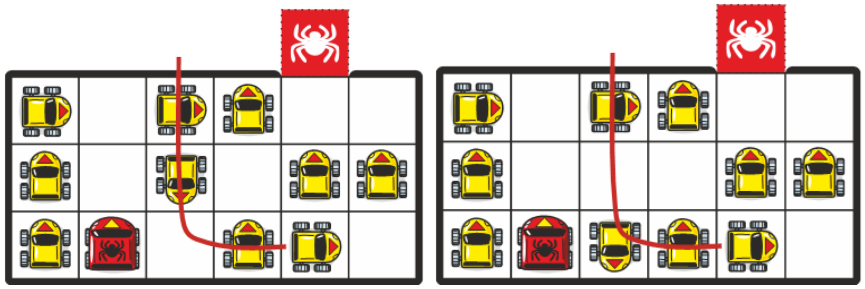
D. 15

Paaiškinimas

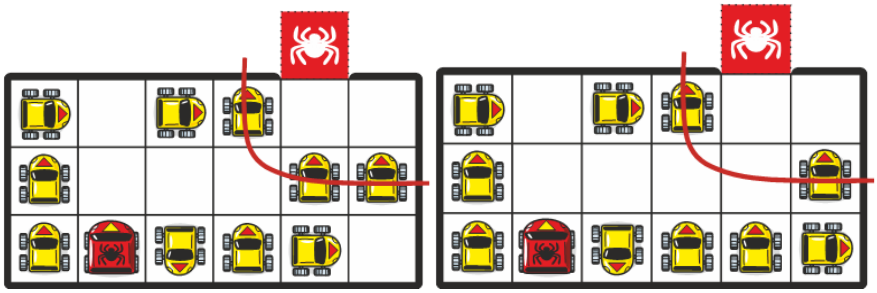
Atsakymas: 11. Automobilis „Voras“ gali pasiekti voro ženklu pažymėtą langelį atlikęs 11 veiksmų, kaip pavaizduota paveiksle:



Belieka įrodyti, kad 11 yra mažiausias skaičius veiksmų šiai užduočiai atlikti. Tarkime, kad automobilis „Voras“ yra vienintelis automobilis aptvertoje teritorijoje. Tam, kad pasiektų voro ženklu pažymėtą langelį jam reikia pajudėti du kartus į viršų ir 3 kartus į dešinę, bei 2 kartus pasisukti. Šie veiksmai gali būti atlikti daugybe įvairių kombinacijų, tačiau automobiliui „Voras“ visuomet prireiks šių 8 veiksmų. Žinoma, automobilis „Voras“ nėra vienintelis aptvertoje teritorijoje ir papildomų veiksmų reikia norint atlaisvinti jam kelią. Pirmiausia turime atlaisvinti kelią pro šią L formos užtvarą. Tai gali būti atlikta atliekant vieną veiksmą:



Dabar turime atlaisvinti kelią pro antrą L formos užtvarą. Tai negali būti atlikta vienu veiksmu neužstatant išvažiavimo. Tam reikia bent dviejų veiksmų:



Taigi, mažiausias automobilių-robotų veiksmų skaičius yra 8+1+2=11.

Tai informatika!

Įrodyti, kad sprendimas yra geriausias (arba optimalus) gali būti labai sudėtinga. Neretai vienintelis būdas tai padaryti yra išvardinti visus sprendimo variantus, norint parodyti, kad nėra jokio geresnio sprendimo už mūsų atrastą. Toks įrodymo būdas yra vadinamas perrinkimo metodu. Dažnai rankiniu būdu patikrinti visų sprendimo variantų gali būti neįmanoma, tačiau tai nesunkiai atliekama naudojant kompiuterį.

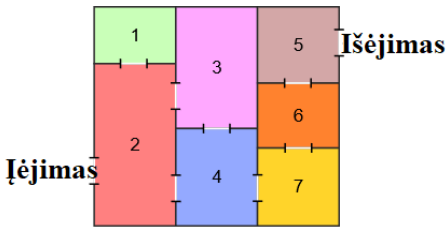
Kartais galimų sprendinių skaičius yra toks didelis, kad net ir kompiuteriu sprendimas, kuriame panaudotas perrinkimo metodas, gali užtrukti per ilgai. Tokiu atveju naudojami kiti algoritmai, supaprastinantys paiešką, pavyzdžiui, godusis algoritmas ar šakų ir apribojimų (angl. *branch and bound*, BnB) algoritmas.

20. Apsilankymas parodoje

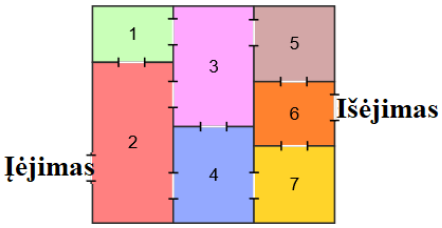
Naujam muziejui pasiūlyti 4 projektai ekspozicijoms įrengti. Kiekvieno projekto brėžinyje yra po 7 sales, kurios sužymėtos numeriais nuo 1 iki 7.

Kuriame projekte yra galimybė būsimos parodos lankytojams apžiūrėti visus eksponatus kiekvienoje iš 7 salių apsilankant tik po vieną kartą?

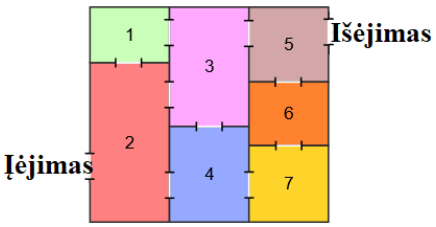
A



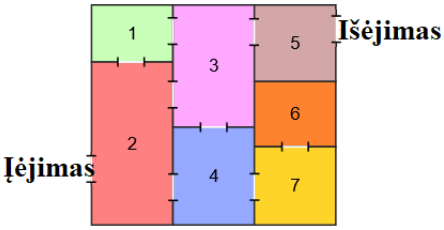
B



C



D



Paaiškinimas

Atsakymas: C.

Tik C projekte numatyta galimybė kiekvienoje salėje apsilankyti tik vieną kartą. Salių seka: 2, 1, 3, 4, 7, 6, 5.

Jei salėje yra tik vienas įėjimas, tai lankytojas, išeidamas iš šios salės, atsidurs salėje, iš kurios įėjo. Todėl ir apsilankys 2 kartus toje pačioje salėje.

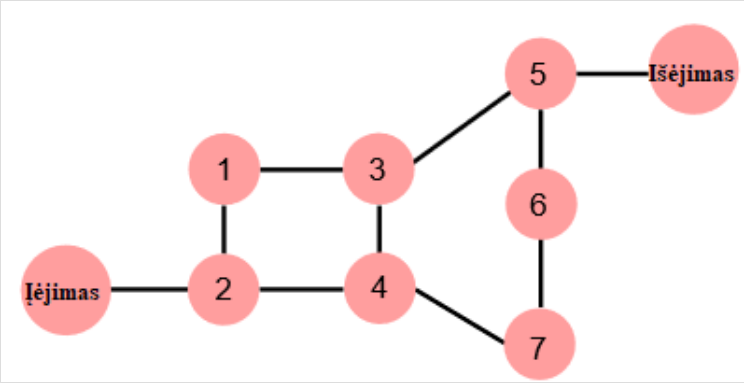
A projekte salėje, kurios numeris 1, yra tik vienas įėjimas, todėl 2 salėje tektų pabuvoti 2 kartus.

B projekte paskutinę 6-ąją salę galima pasiekti iš 5-osios ir 7-osios salės. Aplankius 7 ar 5 salę, 6 salėje teks apsilankyti 2 kartus.

D projekte 6 salė turi tik vieną įėjimą. Ir tai tampa dar didesne problema, nes po 2 kartus tektų apsilankyti net keliose salėse.

Tai informatika!

Norint šio uždavinio sąlygą pateikti kompiuteriui, svarbu schematiškai parodyti, kaip salėse yra išsidėsčiusios durys. Pavyzdžiui, C projektas gali būti konstruojamas taip:

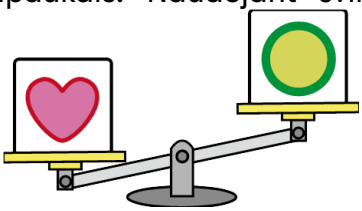


Skrituliukai vaizduoja sales, o atkarpos – įėjimo ir išėjimo vietas. Toks modelis vadinamas grafu. Skrituliukai vadinami grafo viršūnėmis, o atkarpos tarp skrituliukų – briaunomis. Grafai naudojami ieškant trumpiausių maršrutų, ieškant draugų socialiniuose tinkluose.

Šiame modelyje ieškoma tinkamo maršruto tarp visų viršūnių keliaujant briaunomis. Kai salių ir durų kiekiai yra maži, ieškoti sprendinio kompiuterio pagalba nebūtina. Jei salių ir durų būtų labai daug, rasti sprendimą būtų sudėtinga net kompiuteriui.

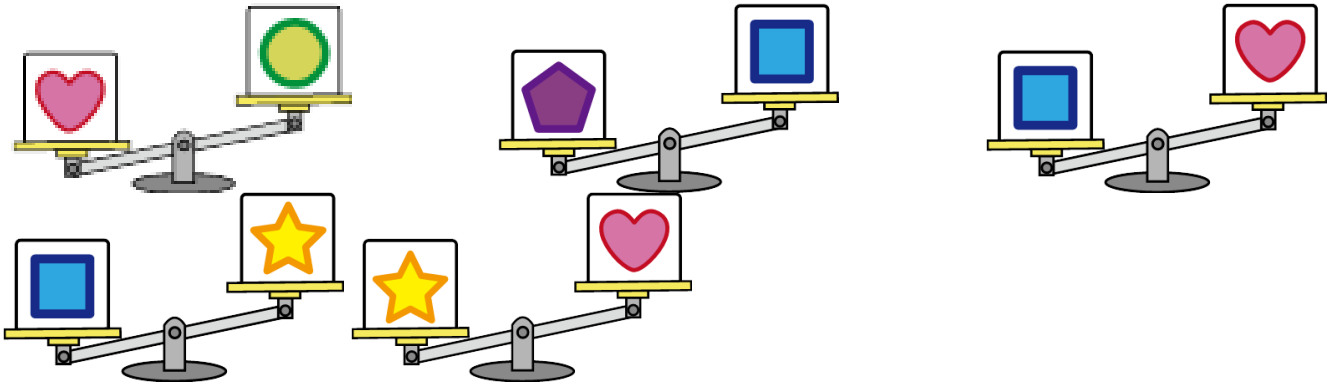
21. Sunkiausia dėžė

Penkios dėžės pažymėtos skirtingais lipdukais. Naudojant svirtines svarstykles galima palyginti dviejų dėžių svorius. Pavyzdžiui:



Matome, kad dėžė su  yra sunkesnė už  .

Atlikti penki svėrimai:



Kuri dėžė sunkiausia?

A. 

B. 

C. 

D. 

E. 

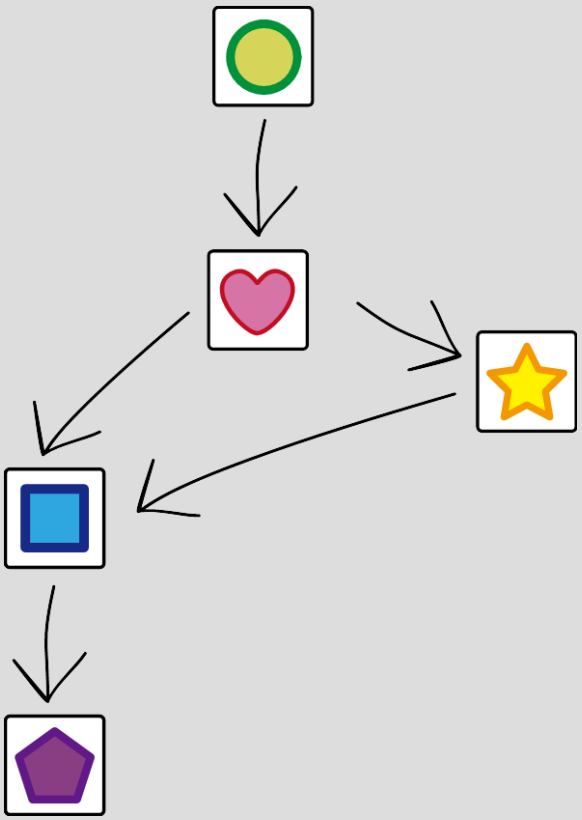
Paaiškinimas

Atsakymas:  (C, dėžė su penkiakampiu).

Ką galime pasakyti apie sunkiausią dėžę? Pirmiausia, ji turi būti sunkesnė už bet kurią kitą dėžę (tai yra, turi būti žemesnėje svarstyklių lėkštėje) atliekant bet kurį svėrimą. Peržiūrėję visus svėrimus, matome, kad dėžė su penkiakampiu vienintelė tenkina šią savybę. Toliau įsitikiname, kad kitos dėžės yra lengvesnės bent viename iš palyginimų, vadinasi, nė viena jų negali būti sunkiausia.

Tai informatika!

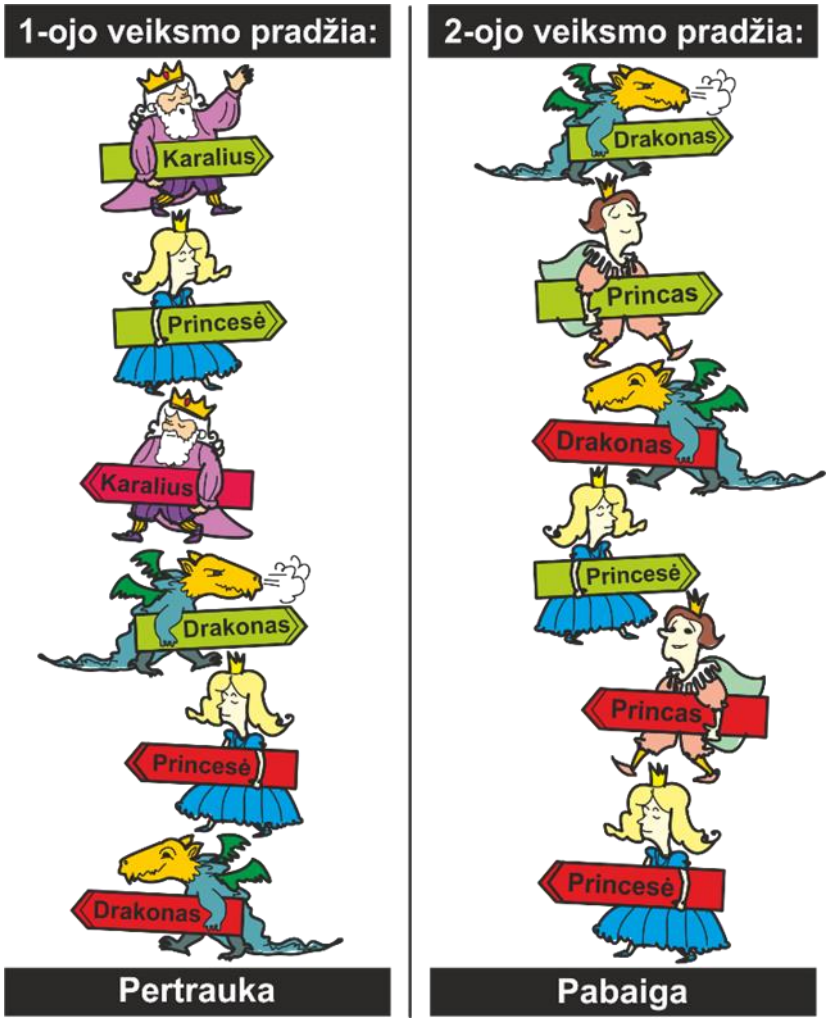
Informatikoje labai dažnai reikia rikiuoti duomenis pagal kuriuos nors požymius, šiame uždavinyje rikiuojamos dėžės pagal svorius. Tokiems uždaviniams spręsti dažnai pasitelkiami grafai – figūra, sudaryta iš taškų (viršūnių) ir linijų (briaunų). Šiame uždavinyje viršūnės yra dėžės, o palyginimai – briaunos. Šio uždavinio grafas atrodytų taip (rodyklė rodo sunkesnę dėžę):



Šitoks rikiavimas vadinamas topologiniu. Kompiuteris lengvai sprendžia tokius uždavinius: einama tiesiog rodyklėmis ir pasiekiamas sunkiausias objektas. Tačiau ne visiems grafams tinka topologinis rikiavimas, pavyzdžiui, jei rodyklės į kelias viršūnes eitu ratu (sudarytų ciklą). Kompiuteris, kaip ir svirtinės svarstyklės, lygina du daiktus. Rikiavimui sukurtų algoritmų efektyvumas matuojamas būtent tokių lyginimų skaičiumi.

22. Teatro vaidinimas

Vaidindami pasaką aktoriai įžengia į sceną ir ją palieka pagal tvarką, parodytą paveiksle (nuo viršaus į apačią). Vaidinimą sudaro du veiksmai ir pertrauka tarp jų.



Rodyklių paaiškinimas:

Aktorius įžengia į sceną

Aktorius palieka sceną

Kuris teiginys neteisingas?

A) Princas ir princesė buvo kartu scenoje.
B) Karalius ir drakonas buvo kartu scenoje.
C) Princas įžengė į sceną po pertraukos.
D) Princas ir drakonas buvo kartu scenoje.

Paaiškinimas

Atsakymas: B.

A teiginys „Princas ir princesė buvo kartu scenoje“ teisingas. Po pertraukos princas įžengia į sceną, o drakonas palieka sceną, tada princesė vėl grįžta į sceną.

B teiginys „Karalius ir drakonas buvo kartu scenoje“ neteisingas, kadangi karalius palieka sceną prieš drakonui įžengiant į sceną.

C teiginys „Princas įžengė į sceną po pertraukos“ yra teisingas.

D teiginys „Princas ir drakonas buvo kartu scenoje“ teisingas. Po pertraukos drakonas įžengia į sceną ir tuoj pat princas ateina į sceną.

Tai informatika!

Informatikoje dažnai procesas vaizduojamas grafiniu būdu (tam tikra tvarka laike vykstantys įvykiai). Informacijos, duomenų ar žinių grafinis pavaizdavimas padeda greičiau suvokti uždavinį, perprasti jo esmę. Žmonės paprastai greičiau supranta problemą ir įžvelgia dėsningumus, jei bent kažkas pateikta grafiškai, schemomis. Taigi suprasti grafinę informaciją ar diagramas, daryti išvadas iš jų yra labai svarbus informatikoje gebėjimas.

Kitas svarbus šio uždavinio komponentas – veiksmų abstrahavimas. Gana sudėtingas teatro procesas pavaizduotas piešiniais – kaip kas kuriuo laiku vyksta. Sprendžiantis uždavinį žmogus, žvelgdamas į piešinius, turi apdoroti daug informacijos: pastebėti rodykles, veikėjus, jų įėjimo į sceną laiką, scenos palikimą, dar atsižvelgti į pertrauką. Abstrahavimas yra labai svarbus gebėjimas sprendžiant informatinio mąstymo problemas ir jo būtina mokytis.

Šį uždavinį galime susieti ir su kintamųjų supratimu: pavyzdžiui, keturi veikėjai (karalius, princesė, princas ir drakonas) gali būti aprašyti kintamaisiais, kurie įgyja reikšmes „įžengia į sceną“ arba „palieka sceną“ arba tik logines reikšmes „taip“ ir „ne“, atsakančias į klausimą „Ar veikėjas yra scenoje?“

	Nykštukai	Mažyliai	Bičiuliai	Draugai	Jauniai	Kolegos
				12	9	6

23. Apgadinta lentelė

Slaptas rašymas gali būti pagrįstas lotyniškos abėcėlės raides koduojant naujais simboliais. Kaip koduojama, aprašo pateikta lentelė. Deja, dalis lentelės informacijos nutrinta.

Nors lentelė apgadinta, vis dėlto galima bandyti iškoduoti tekstus.

Iškoduokite frazę:

Kuris atsakymo variantas tinkamas?

A. INFORMATION SECRET

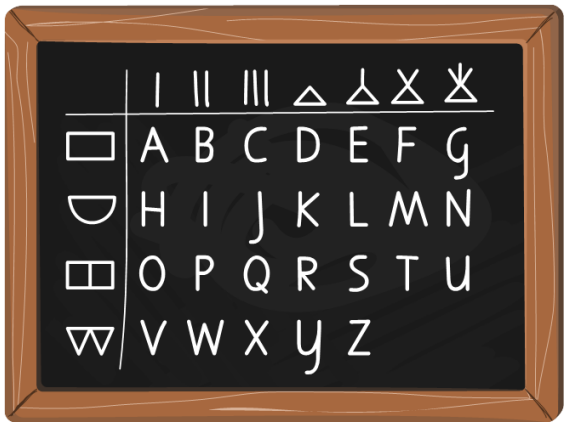
B. INFORMATICS IS COOL

C. MATHEMATICS IS COOL

D. INFORMATION IS COOL

Paaiškinimas

Visa kodavimo lentelė atrodo šitaip:



Atsakymas yra: INFORMATICS IS COOL.

Norėdami išspręsti šį uždavinį turime pastebėti, kad nauji simboliai sudaromi kombinuojant eilučių ir stulpelių žymenis. Eilutės žymuo sudaro apatinę simbolio dalį, o stulpelio – viršutinę. Išsiaiškinus tai, lengva atstatyti trūkstamus simbolius. Žinoma, reikia žinoti lotynų abėcėlę. Pirmai eilutei trūksta žymens, kurį galima nesunkiai nustatyti iš pateikto užkoduoto teksto apžiūrėjus apatines simbolių dalis. Kai kurias raides jau žinome, kaip jas koduoti. Įstatome jas į užkoduotą frazę.

I N _ O _ _ _ _ I _ I S _ _ O L L

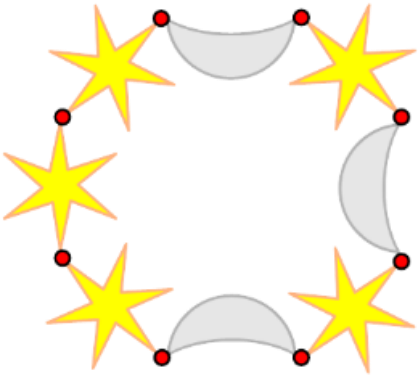
Toliau bandymų ir klaidų metodu galima ieškoti trūkstamų raidžių. Tačiau esant pateiktiems variantams, net to nereikia.

Alternatyviai galima spręsti net neatstačius lentelės. Pakanka pastebėti, kad pirmasis simbolis atitinka raidę „I“, todėl C atsakymo variantas netinka. Vienuoliktas simbolis atitinka raidę „S“, tačiau A ir D atvejais tai yra kita raidė. Taigi belieka vienintelis teisingas atsakymas – B variantas.

Tai informatika!

Duomenų saugumu imta rūpinti prieš 4000 metų. Kriptografija yra viena svarbiausių informatikos sričių, ji skirta slaptiems kodams kurti, kad galima būtų apsaugoti konfidencialius duomenis ar pranešimus. Senosios kultūros naudojo paprastus slapto rašymo būdus, kurių aprašymas turėjo būti lengvai įsimenamas. Šiame uždavinyje ir pateikiamas tokio senovinio slapto rašymo pavyzdys. Tai yra paprastas šifravimas, kai kiekviena abėcėlės raidė keičiama vienu pasirinktu simboliu.

24. Žvaigždės ir pusmėnuliai



Marija nori piešinyje pavaizduotos apyrankės.

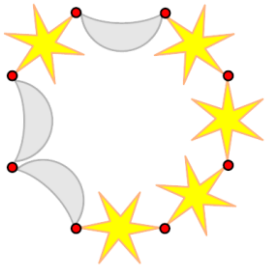
Ji pateikia instrukciją Jonui, kaip pagaminti tokią apyrankę:

- Paimk vieną žvaigždę ( ) ir vieną pusmėnulį ( ) ir juos sukabink.
- Pakartok pirmą žingsnį dar du kartus, – turi gauti tris poras.
- Gautas tris poras sujunk į grandinę.
- Prikabink dvi žvaigždes prie vieno iš grandinės galų, tada galus sukabink – apyrankė baigta!

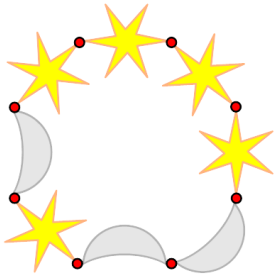
Jei Jonas neturėtų apyrankės piešinio, tai jo gaminama apyrankė galėtų gerokai skirtis, nors jis viską atliktų pagal duotą instrukciją.

Jonas pagal Marijos instrukciją galėtų pagaminti tris iš parodytų apyrankių. Kurios apyrankės Jonas negalėtų pagaminti?

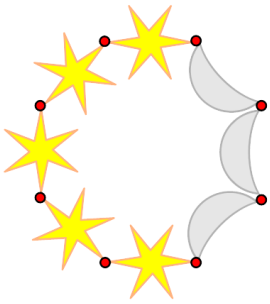
A.



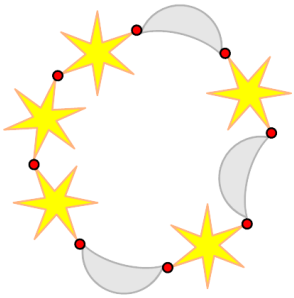
B.



C.



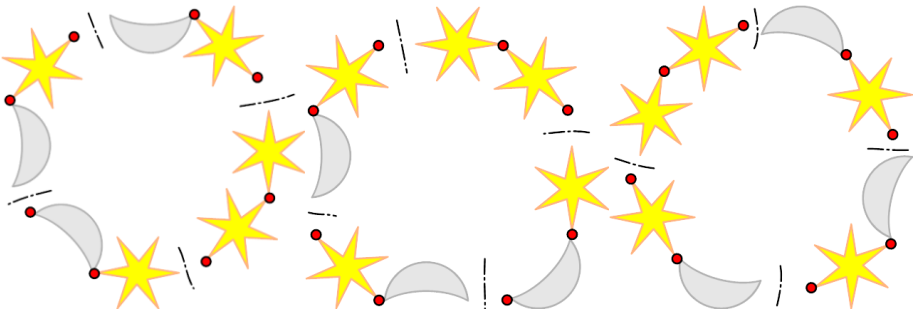
D.



Paaiškinimas

Atsakymas: C.

Paveikslėliuose parodyta, kaip kiekvieną iš trijų likusių apyrankių galima padalyti į tris „žvaigždė–pusmėnulis“ poras ir dar vieną porą „žvaigždė–žvaigždė“.



Kadangi šalia kiekvieno pusmėnulio turi būti po žvaigždę (pirma instrukcijos taisyklė), tai trys iš eilės pusmėnuliai negali būti. Būtent C atveju yra trys pusmėnuliai.

Tai informatika!

Kai programuotojas pateikia instrukcijas kompiuteriui, labai svarbu tiksliai nusakyti, ką kompiuteris turi atlikti, kitaip galima gauti ne tą rezultatą, kurio tikėjosi. Pavyzdžiui, rašydama instrukciją, Marija pamiršo nurodyti, kaip trys poros „žvaigždė–pusmėnulis“ turi būti sujungtos tarpusavyje. Piešinyje matome, kad Marija norėjo, jog prie abiejų pusmėnulio galų būtų prikabinta po žvaigždę. Taigi, nors instrukcijos aprašas atrodo gana aiškus, vis tik dažnai būna, kad kažkas pamirštama, nepasakoma. Jei turėtume apyrankes gaminantį kompiuterį, Marijos instrukcijos nepakaktų. Laimei, realybėje, jei instrukcija nėra pakankamai aiški, kompiuteris sustoja ir klausia, ką toliau daryti.

Informatikoje yra priemonių instrukcijoms aprašyti – vadinamosios formaliosios specifikacijos. Viena iš dažniausiai taikomų formaliųjų specifikacijų yra gramatika. Gramatiką sudaro taisyklės, kaip iš vienų žodžių gauti kitus, ir patys žodžiai ar simboliai. Marijos instrukciją galėtume užrašyti štai tokia gramatika:

P → ŽM (1)

K → PPP (2)

A → KŽŽ (3)

Čia A reiškia apyrankę, P – porą, Ž – žvaigždę ir M – pusmėnulį. Ši gramatika tiksliai aprašo Marijos norimas apyrankes, jokių papildomų paaiškinimų nereikia. Pavyzdžiui:

A ⇒ KŽŽ (3)

KŽŽ ⇒ PPPŽŽ (2)

PPPŽŽ ⇒ ŽMPPŽŽ ⇒ ŽMŽMPŽŽ ⇒ ŽMŽMŽMŽŽ (1)

25. Bebras pilyje

Sumanus bebras nori užtvenkti upelį ir todėl jam reikia eglės. Tačiau bebras turi tik morką. Šiandien pilyje mugė, kurios metu galima apsikeisti įvairiais daiktais. Bebras skuba į pilį su morka, kad galėtų ją išmainyti į eglę.

Kiekvienoje pilies menėje galimi tik tokie mainai:

Menėje A: → arba →

Menėje B: → arba →

Menėje C: → arba →

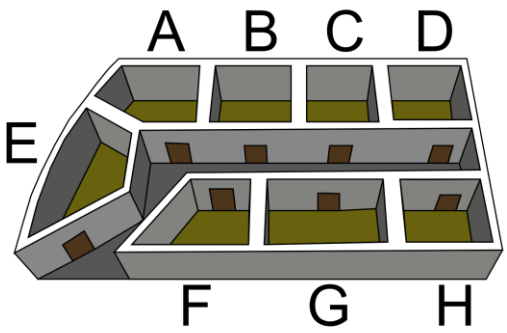
Menėje D: → arba →

Menėje E: → arba →

Menėje F: → arba →

Menėje G: → arba →

Menėje H: → arba →



Kuriose menėse ir kokia eilės tvarka turi apsilankyti bebras, kad jam pavyktų morką išmainyti į eglę?

A. DGE

B. GGE

C. AGE

D. DBC

Paaiškinimas

Atsakymas: DGE.

Menėje D bebras morką išmaino į ledus . Tada jis eina į menę G ir ledus išmaino į žiedą . Po to jis ateina į menę E ir žiedą išmaino į eglę .

Menėje D: →

Menėje G: →

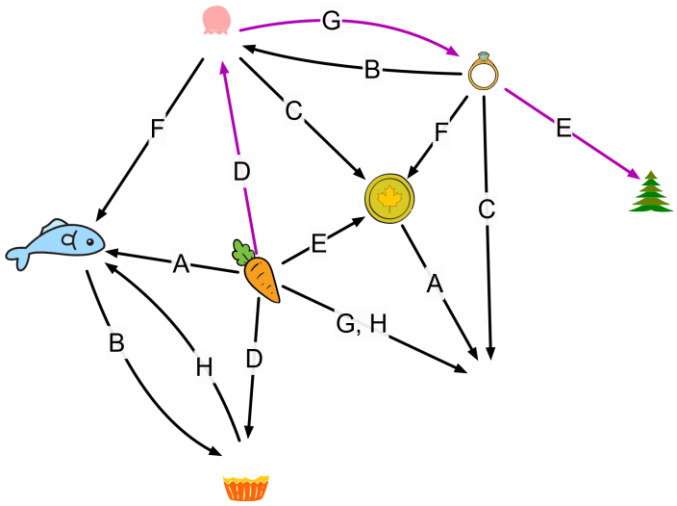
Menėje E: →

Sprendžiant šią užduotį, galimos dvi strategijos.

Pirmos strategijos idėja – bebras morką gali išmainyti penkiose menėse (A, D, E, G ir H) rinkdamasis iš šešių objektų. Nagrinėjant visas 6 galimybes įmanomas net begalinis ciklas.

Pavyzdžiui:

1. Menėje A (→)
2. Menėje B (→)
3. Menėje H (→)
4. Menėje B (→) – gauname ciklą!



Antroji strategija efektyvesnė ir remiasi įvykių inversijos principu. Pradedame nuo rezultato, kai menėje E yra galimybė žiedą išmainyti į eglę . Menėje E ir yra vienintelė galimybė bebrui mainais gauti eglę. Atitinkamai, tik menėje G galima gauti žiedą mainant į jį ledus . Išmainyti morką į ledus galima tik vienintelėje menėje D.

Mainų galimybes galima pavaizduoti grafu. Kiekviena viršūnė vaizduoja mainomąjį objektą, o briauna – menę, kurioje galimi mainai.

Užduoties vaizdavimas grafu padeda vaizdžiau pamatyti sąryšį tarp morkos ir eglės . Taip pat lengviau įsitikinama, kad galimas tik vienas teisingas atsakymas DGE.

Tai informatika!

Šią užduotį galima pavaizduoti grafu. Informatikoje orientuotasis grafas naudojamas užduoties sąlygai vizualizuoti, demonstruojant galutinę mašinos būseną. Grafų naudojimas teikia pranašumą sprendžiant tokio tipo užduotis. Keliaujant žingsnis po žingsnio grafu galima sudaryti sisteminę seką veiksmų – algoritmą. Pavyzdžiui, gali būti naudojamas paieškos į gylį (angl. *depth-first search*, DFS) algoritmas, kai pasirinkus pradinę viršūnę einama grafo briaunomis kiek įmanoma giliau, renkantis vis naują viršūnę. Kai paskutinė aplankyta viršūnė naujos (dar neaplankytos) kaimynės nebeturi, tada grįžtama iki artimiausios neaplankytos briaunos ir vėl ieškoma kuo giliau tol, kol bus rastas ieškomas tikslas arba kol bus aplankytos visos grafo viršūnės ir briaunos.

Šiuo atveju algoritmas prasidėtų morkos mainais, kol būtų surasta eglė. Šis algoritmas reikalauja patikrinti, ar nebus begalinis ciklas (pavyzdžiui, ABHBHBH...).

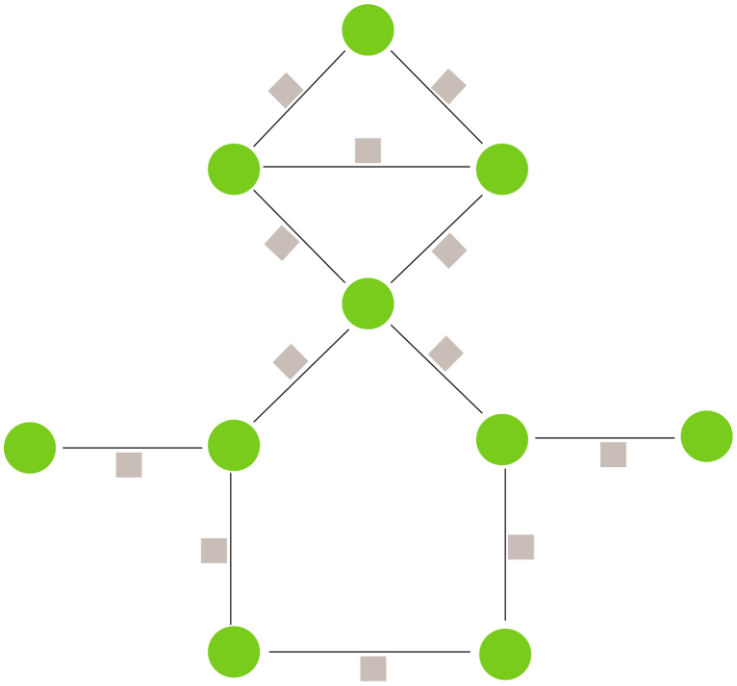
	Nykštukai	Mažyliai	Bičiuliai	Draugai	Jauniai	Kolegos
				12		6

26. Maisto prekių parduotuvės

Dėl pandemijos dauguma parduotuvių uždaroma siekiant sumažinti sveikatai keliamą riziką. Bet šalia kiekvieno kaimo turi būti bent viena maisto parduotuvė. Laikoma, kad parduotuvė yra šalia kaimo, jei pakeliui iš to kaimo į parduotuvę nepraeinama pro jokį kitą kaimą.

Žali skrituliai žymi kaimus, kvadratėliai – parduotuves. Linijos – tai gatvės, jungiančios kaimus. Veikiančios parduotuvės žymimos raudonais kvadratėliais.

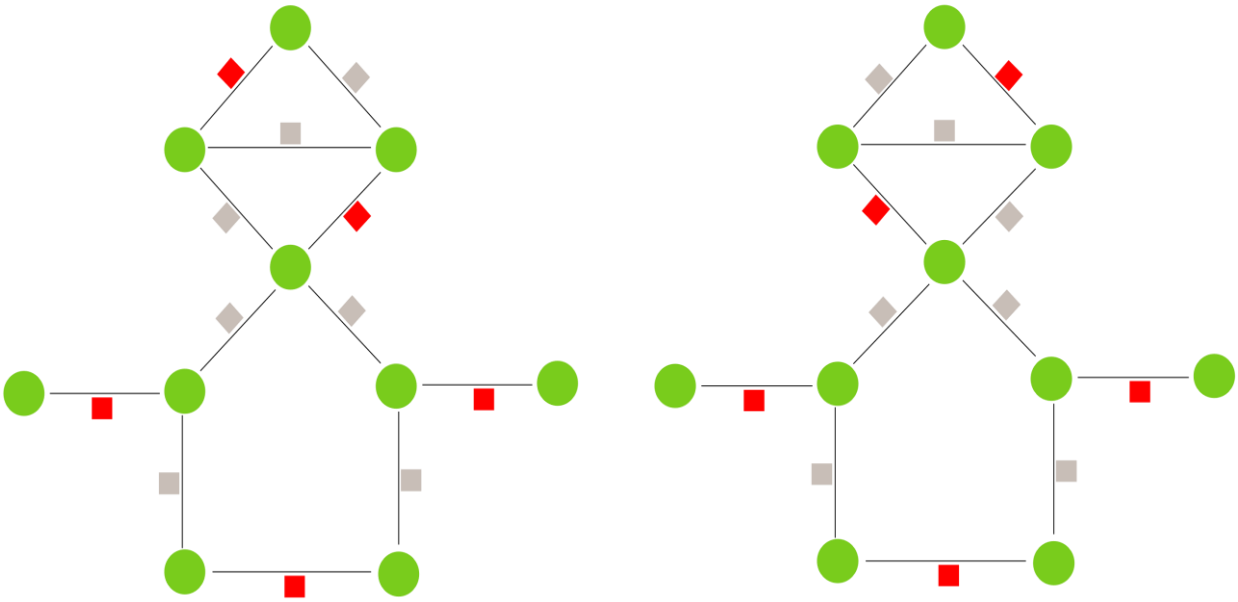
Koks mažiausias žemėlapyje parodytų parduotuvių skaičius turi veikti siekiant užtikrinti aprūpinimą maistu?



Paaiškinimas

Turi likti neuždarytos 5 maisto parduotuvės.

Labiausiai kairėje ir dešinėje esančios parduotuvės turi dirbti, kad aprūpintų schemoje pavaizduotus labiausiai kairėje ir dešinėje esančius kaimus. Kad būtų aprūpinti maistu du kaimai, esantys schemos apačioje, pakanka, kad viena schemos apačioje esanti parduotuvė būtų neuždaryta. Taip pasirinkome 3 parduotuves ir išnagrinėjome kaimus, išskyrus keturis kaimus schemos viršuje. Schemos viduryje esanti parduotuvė nesujungta su kaimais schemos viršuje ir apačioje, taigi turime pasirinkti dar bent dvi parduotuves. Pasirinkę dvi viršutinėje schemos dalyje priešais esančias parduotuves (nesvarbu, kurias), rasime sprendinį – 5 parduotuves. Tai labiausiai į kairę ir labiausiai į dešinę nutolusios parduotuvės, viena schemos apačioje, ir dvi viršutinėje schemos dalyje priešingose kvadrato kraštinėse.



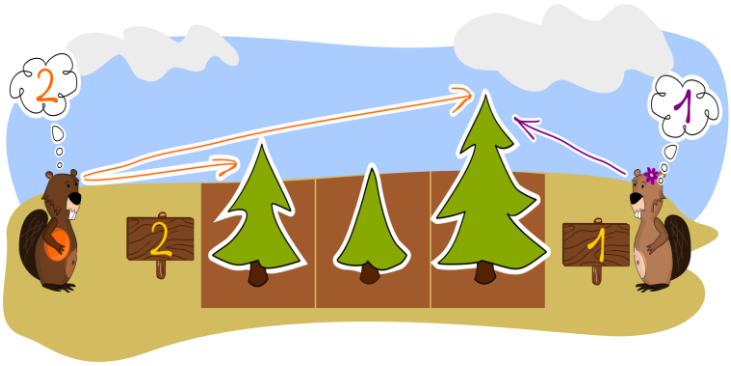
Tai informatika!

Šioje užduotyje magrinėjama grafo mažiausio lankų skaičiaus radimo problema. Tikslas – rasti mažiausią lankų skaičių, kad kiekviena grafo viršūnė būtų „apimta“ vienu iš pasirinktų lankų. Čia kaimai – tai grafo viršūnės, kurias reikia apimti, gatvės su maisto parduotuvėmis – lankai, jungiantys viršūnes.

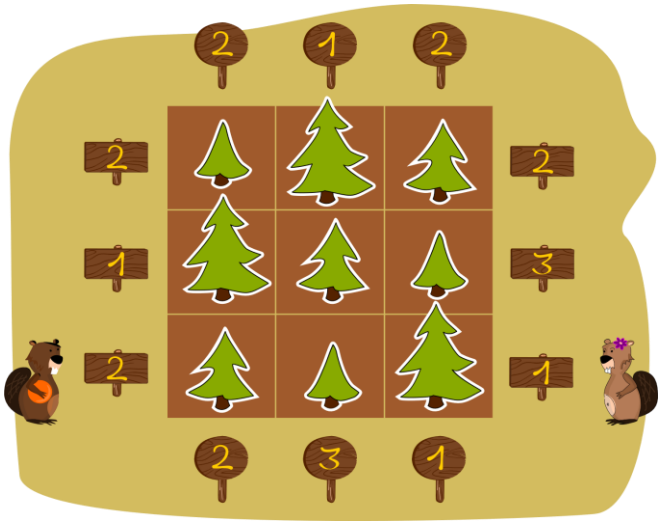
Tokią situaciją galime rasti daugelyje žinomų informatikos uždavinių. Pavyzdžiai: minimalus jungusis medis, naudojamas Dijkstros algoritme trumpiausiam keliui grafe rasti, keliaujančio pirklio uždaviniui spręsti.

27. Medžių sudoku

Lauke bebrai sodina medžius. Medžiai yra trijų aukščių: 1 (), 2 (), arba 3 (). Kiekvienoje eilėje – horizontaliai ir vertikaliai - sodinama tiksliai po vieną kiekvieno aukščio medį. Bebrai eina aplink lauką ir skydelyje ties kiekviena eile pažymi, kiek joje mato medžių: aukštesni medžiai užstoja žemesnius ir jų nematyti (žr. paveikslą).

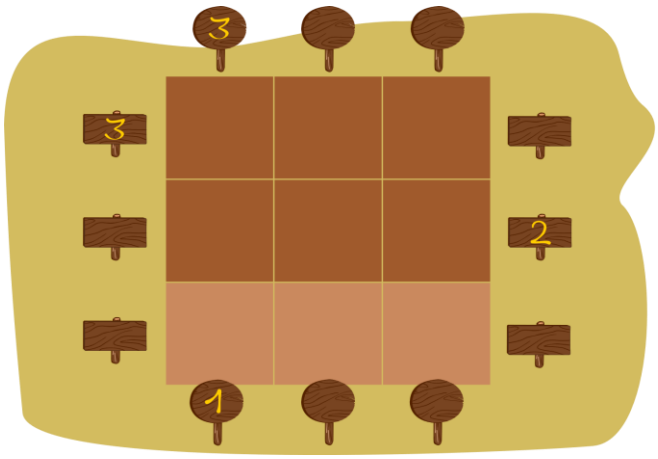


Štai kaip atrodytų devyniais medžiais užsodintas 3x3 eilių laukas:



Kuris iš pateiktų variantų tinka paskutinei (šviesesnės spalvos) eilei užpildyti, kad būtų teisingi skydeliuose įrašyti skaičiai? (Pildoma iš kairės į dešinę.)

A. 1 (), 2 () ir 3 ()
B. 3 (), 1 () ir 2 ()
C. 3 (), 3 () ir 1 ()
D. 3 (), 2 () ir 1 ()

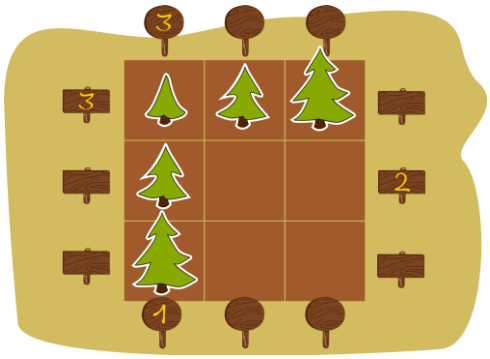


Paaiškinimas

B – vienintelis galimas sprendimas.

C variantą galima iš karto atmesti, nes eilėje negali būti dviejų vienodo aukščio medžių.

Matome, kad dviejuose skydeliuose įrašyta po skaičių 3, vadinasi, iš čia matomi trys medžiai. Tai įmanoma tik tada, kai medžiai išrikiuoti nuo žemiausio iki aukščiausio: . Taigi kairioji ir viršutinės eilės nustatytos:

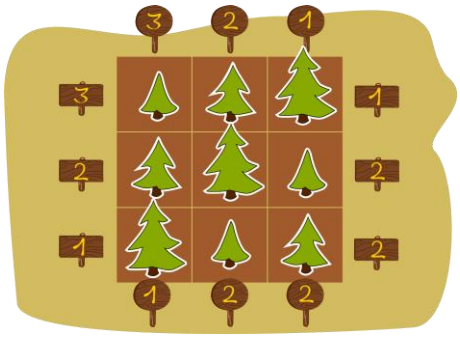


Gauname, kad A atsakymas netinka, kadangi pirmojo medžio aukštis nėra lygus 1.

Toliau medžius sodiname pagal sudoku taisyklę: kiekvienoje eilėje turi būti tik po vieną vieno aukščio medį. Dešinėje skydelyje įrašytas skaičius 2 rodo, kad šioje eilėje matomi du medžiai, vadinasi, viduryje turi būti medis, kurio aukštis 3 , tad gauname visą vidurinę eilę: 2(), 3(), 1().

Paskutinė eilė gaunama automatiškai: jau turime kairiausią medį, kurio aukštis 3 (), toliau tegali būti 1(), nes kiti aukščiai jau panaudoti eilėje vertikaliai, tad paskutinis medis yra 2().

Vienintelis galimas sprendimas:



Tai informatika!

Šiuo uždaviniu ugdomos trys esminės informatikos kompetencijos.

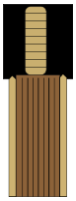
Pirmoji – tai gebėjimas rasti sprendimą pagal duotus ribojimus arba tenkinant nurodytas sąlygas.

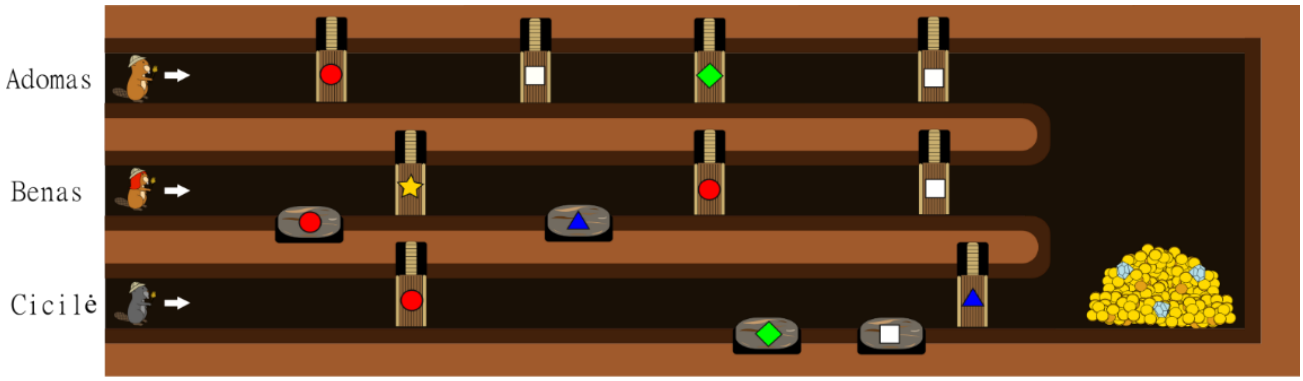
Antroji kompetencija – gebėjimas rekonstruoti objektą iš dalinės informacijos panaudojant duotas objekto savybes. Visa tai gali būti panaudojama suglaudintiems objektams vaizduoti. Konceptas „glaudinimas neprarandant“ vartojamas informatikoje aprašyti procesui, kai glaudinant vartotojas nepraranda jokios informacijos, kuri buvo esant nesuglaudintam pavidalui.

Trečioji kompetencija susijusi su programavimu, tiksliau, testavimu ir verifikavimu. Skydeliuose nurodomi skaičiai yra pavyzdys, kaip galima automatiškai aptikti klaidas.

28. Ieškome lobio

Jaunieji tyrinėtojai Adomas, Benas ir Cecilė ieško lobio, paslėpto slaptame kambaryje. Kiekvienas tyrinėtojas eina skirtingu keliu.

Kiekviename kelyje gali pasitaikyti pakeliamų durų  ir įspaudžiamų akmenų  . Durys ir akmenys pažymėti figūromis. Visos durys iš pradžių yra užrakintos. Tyrinėtojas, priėjęs užrakintas duris, turi sustoti ir laukti, kol durys atsirakins. Tyrinėtojui užmynus ir įspaudus akmenį, atrakinamos visos durys su tokia pačia figūra, kaip ir įspausto akmens. Paveikslėlyje matyti, kaip durys ir akmenys išsidėstę kiekvieno tyrinėtojo kelyje:



Kas pasieks slaptąjį kambarį?

A) Adomas

B) Benas

C) Cecilė

D) Niekas nepasieks slaptojo kambario

Paaiškinimas

Atsakymas: A.

Atsakymas B neteisingas, nes niekur nėra akmens su geltona žvaigžde, tad Benas savo kelyje sustos prie pirmųjų durų.

Atsakymas C neteisingas, nes paskutinės durys Cecilės kelyje atsirakintų tik jei Benas užmintų akmenį, pažymėtą mėlynu trikampiu. Bet pagal atsakymo B paaiškinimą, Benas niekada neprieis iki to akmens.

Teisingas yra atsakymas A. Benas atrakins duris, pažymėtas raudonu skrituliu. Po to Cecilė atrakins visas duris, pažymėtas žaliais rombais ir baltais kvadratais. Tuomet visos durys Adomo kelyje bus atrakintos ir Adomas galės patekti į slaptąjį kambarį.

Atsakymas D neteisingas, nes slaptąjį kambarį pasieks Adomas.

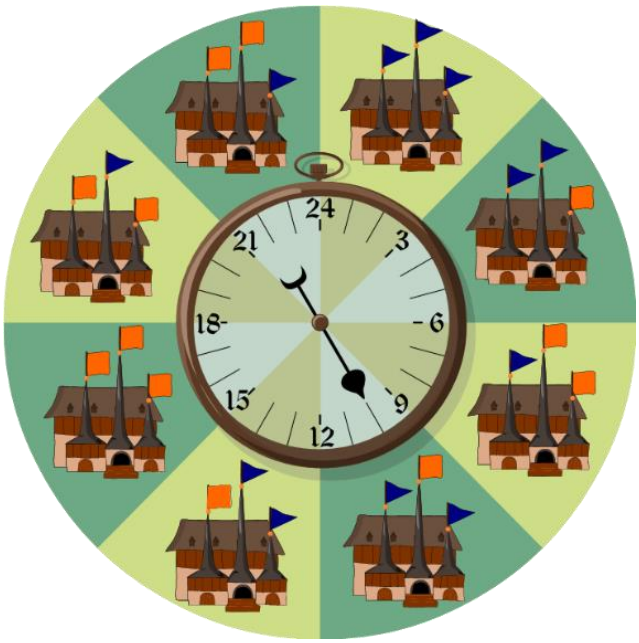
Tai informatika!

Tris jaunuosius tyrinėtojus šiame uždavinyje galima palyginti su trimis kompiuteriais (arba procesoriais), veikiančiais lygiagrečiai ir naudojančiais bendrą blokuočių mechanizmą. Kiekvienas kompiuteris gali panaikinti blokuotę (kaip tyrinėtojas užminti įspaudžiamą akmenį) arba pristabdyti darbą, kol blokuotė nepanaikinta (kaip tyrinėtojas prieina užrakintas duris).

Pastaruoju metu dėl ribotų technologinių galimybių tampa vis sunkiau padidinti kompiuterio procesoriaus spartą, todėl į kompiuterius imta dėti po kelis procesorius (branduolius).

Kad išnaudotų keleto lygiagrečiai veikiančių procesorių teikiamus privalumus ir tokiu būdu paspartintų kompiuterio programų veikimą, kompiuterius kuriantys mokslininkai turi rasti būdą, kaip skaičiavimus išskaidyti į atskiras dalis, kurios galėtų būti apdorojamos lygiagrečiai, t. y., mokslininkai turi sukurti lygiagrečiuosius algoritmus. Kartais skaičiavimus viename procese reikia pristabdyti, kol bus baigti skaičiavimai kitame, todėl tokiems skaičiavimams naudojamas blokuočių mechanizmas – jis neleidžia vienam procesui tęsti skaičiavimų, kol kitas procesas nepanaikins blokuotės.

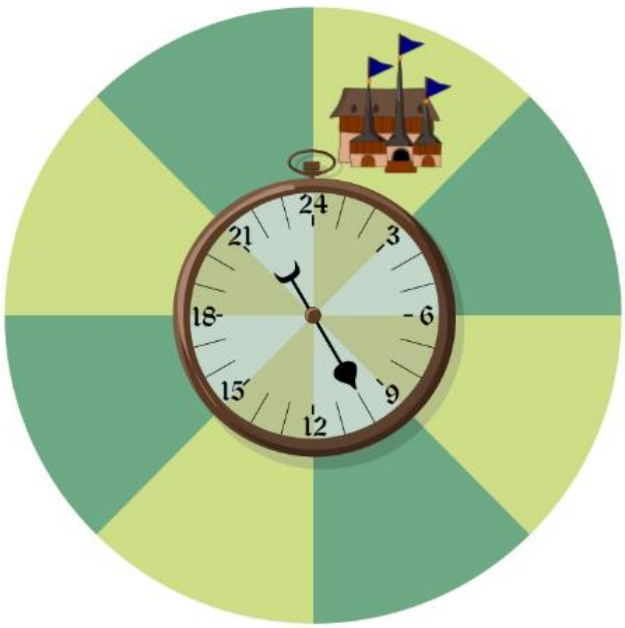
29. Bebrų miestelio laikrodis

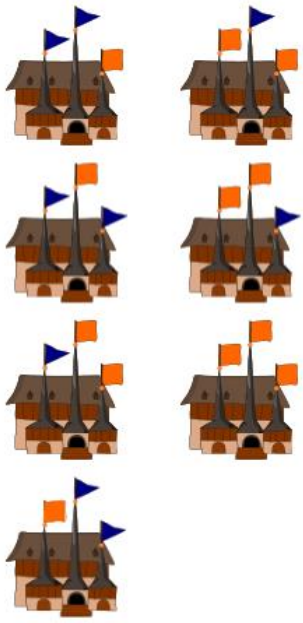


Keistame mažame bebrų miestelyje diena yra dalinama į 8 intervalus po 3 valandas. Jo gyventojai bebrai nori žinoti tik dabartinį laiko intervalą. Norėdami tai padaryti, jie pasižiūri į tris rotušės vėliavas ir pagal vėliavų kombinaciją nustato, kuris tai laiko intervalas. Kiekviena iš šių vėliavų gali būti arba mėlynas trikampis, arba oranžinis kvadratas.

Ši sistema yra pusiau tobula, nes norint pakeisti laiką pirmuose keturiuose intervaluose tereikia perkabinti vieną vėliavą. Tačiau norint pakeisti laiką kituose intervaluose bebrams tenka perkabinti daugiau nei vieną vėliavą.

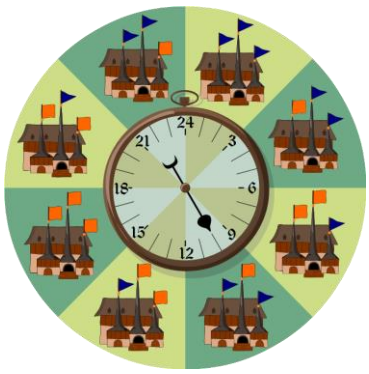
Pasiūlykite tobulą vėliavų sistemą, kurioje, norint pakeisti laiką, visuomet užtektų perkabinti tik vieną vėliavą. Yra daug galimų tobulų sistemų variantų – pateikite bet kurį iš jų.





Paaiškinimas

Vienas iš daugybės galimų sprendinių:



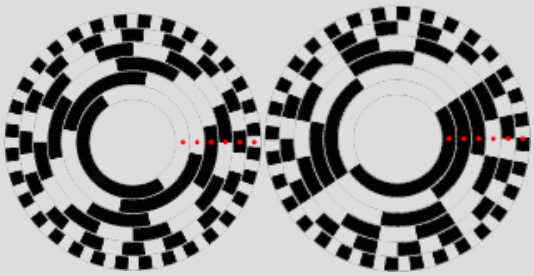
Naudokime dvejetainę vėliavų būsenų reprezentaciją – 0 mėlynam trikampiui pažymėti ir 1 raudonam kvadratui pažymėti. Tokiu atveju turime 8 skirtingas trijų vėliavų kombinacijas: 000, 001, 010, 011, 100, 101, 110, 111. Šiuos skaičius mums reikia išdėlioti ratu taip, kad bet kurie du greta esantys skaičiai skirtųsi tik vienu skaitmeniu.

Pradėkime nuo dviejų skaičių, pasižyminčių šia savybe: 000 ir 100. Tuomet pridedame 110 (arba 101) iš dešinės ir gauname 000, 100, 110. Tuomet galime pridėti 010 ir gauti 000, 100, 110, 010. Vienintelis kitas galimas skaičius yra 011. Po jo pridedame 111 ir gauname 000, 100, 110, 010, 011, 111. Tėra du likę skaičiai: 001 ir 101. Antrasis vienu skaitmeniu skiriasi nuo 111, o pirmasis vienu skaitmeniu skiriasi nuo 000. Taigi gauname 000, 100, 110, 010, 011, 111, 101, 001. Ši sistema tobula. Kadangi mes galėjome skirtingai pasirinkti pirmąją porą, o vėliau – trečiąjį skaičių, yra daug galimų sprendimo variantų.

Tai informatika!

Toks skaičių išdėstymas, vadinamas Grėjaus kodu, turi daugybę pritaikymo sričių. Tai, kad tik vieno bito pakeitimas gali pakeisti užkoduotą žodį į greta esantį, pavyzdžiui, gali sutaupyti procesui vykdyti naudojamos energijos. Kelių bitų reikšmių pakeitimas reikalauja daugiau energijos, nei vieno bito, o įprastame dvejetainiame kode du gretimi skaičiai neretai skiriasi daugiau, nei vienu bitu.

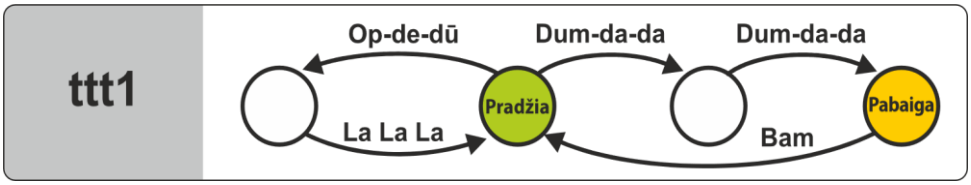
Žymus Grėjaus kodo pritaikymas inžinerijoje yra besisukančio disko kampo matavimas. Kairiajame paveiksliuke Grėjaus kodas pavaizduotas ant disko. Virš disko pritaisyta eilė šviesos jutiklių (paveikslėlyje pavaizduota raudonais taškeliais), galinčių atskirti juodą ir baltą spalvas, leidžia mums nuskaityti dvejetainį skaičių, reiškiantį disko sukimosi kampą.



Kairiajame paveikslėlyje galime matyti, kaip šviesos jutikliai perskaito arba kodą 001010 arba kodą 001110, nes ketvirtasis jutiklis yra ties baltos ir juodos spalvos riba. Bet kurio atveju tokia situacija nėra pavojinga, nes tikrasis kampas yra tarp kampų, užkoduotų gretimais kodais 001010 ir 001110. Naudojant kitokį kodą tokia situacija galėtų būti net labai pavojinga. Pavyzdžiui naudojant įprastą dvejetainį kodą, kaip dešiniajame paveikslėlyje, greta atsiduria kodai 111010 ir 111001. Jei diskas būtų pasisukęs tarp šių dviejų pozicijų, jutikliai galėtų klaidingai perskaityti 111011, kuriuo užkoduotas kitas, pakankamai nutolęs sukimosi kampas. Blogiausiu atveju, jutikliai turėtų kirsti kampą tarp 000000 ir 111111. Šiuo atveju visi jutikliai galėtų rodyti bet kurią iš būsenų 0 arba 1, taigi – bet kurį iš galimų sukimosi kampų!

30. Laimos dainelės

Bebrė Laima mėgsta dainuoti. Ne bet kokias dainas, o tik „gražias daineles“. Tokios dainos gali būti atliekamos pagal **ttt1** schemą:



Laima pradeda dainuoti vieną iš savo dainų nuo apskritimo, pažymėto „Pradžia“ ir naudojasi rodyklėmis išdainuodama ant jų užrašytus skiemenis. Baigia tik tada, kai pasiekia „Pabaiga“ (tačiau ji gali ir nesustoti pasiekusi „Pabaiga“).

Galimos dainos:

Op-de-dū La La La Op-de-dū La La La

Dum-da-da Dum-da-da Bam Dum-da-da Dum-da-da

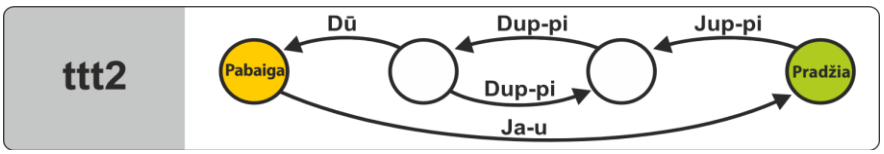
arba

Dum-da-da Dum-da-da Bam Op-de-dū La La La

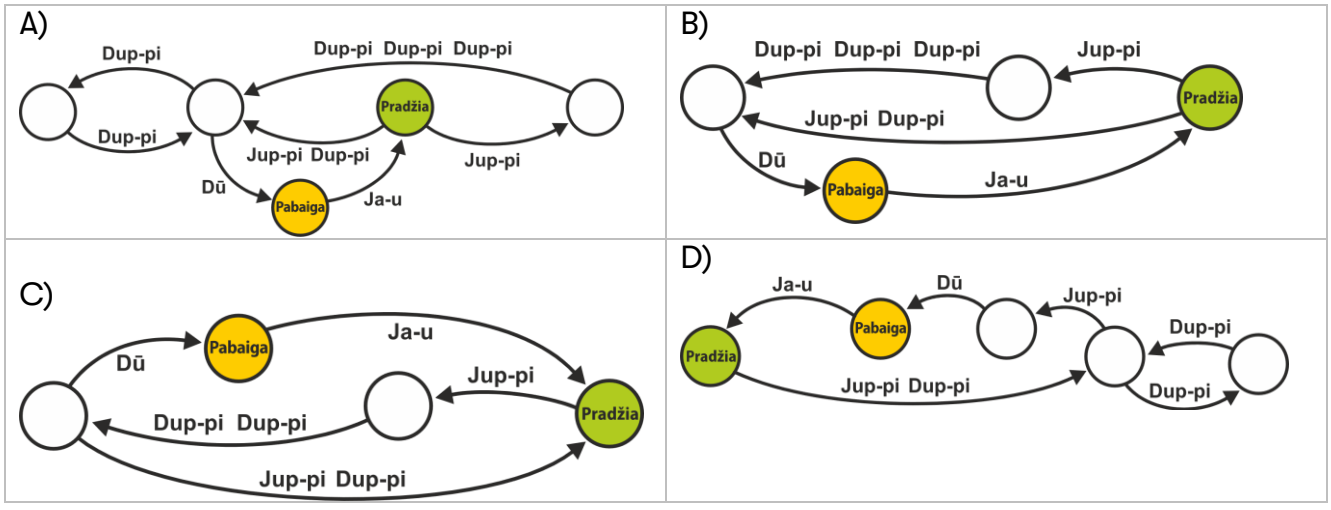
Dum-da-da Dum-da-da Bam Op-de-dū La La La

Dum-da-da Dum-da-da Bam Dum-da-da Dum-da-da

Naujųjų metų šventei Laima ruošiasi dainuoti pagal naują schemą (**ttt2**):



Kurią iš šių keturių schemų galima naudoti toms pačioms dainoms dainuoti, kaip ir pagal **ttt2** schemą?



Paaiškinimas

Atsakymas: A.

Dainuojant pagal ttt2 diagramą, dainelė visada prasideda „Ju-pi“. Toliau bent vieną kartą eina „Dup-pi“. Tada prisideda „Dū“ arba lyginis skaičius „Dup-pi“ ir „Doo“. Čia gali būti arba dainos pabaiga, arba prisidėti „Ja-u“ ir viskas prasidėti iš naujo.

Pirmoji atsakymų variantų diagrama tinka toms pačioms dainoms, kaip ir duotoji ttt2. Ji taip pat pradeda dainas nuo „Jup-pi“. Tai gali būti „Jup-pi Dup-pi“ arba „Jup-pi Dup-pi Dup-pi Dup-pi“. Tada lyginį skaičių kartų gali būti kartojamas „Dup-pi“ ir „Dū“ dainos pabaigoje. Kaip ir ttt2, po „Ja-u“ galima pradėti iš naujo.

B ir C diagramose negali būti kartojamas nelygins skiemenų „Dup-pi“ skaičius, todėl yra ttt2 dainų, kurių negalime generuoti B ir C diagramomis. Šie skiemenys pasirodo vieną ar tris kartus, tačiau ne penkis kartus. D diagrama leidžia generuoti nelyginį skaičių „Dup-pi“, bet prieš paskutinį „Dū“ gali pridėti „Jup-pi“, kurio ttt2 diagrama negali generuoti.

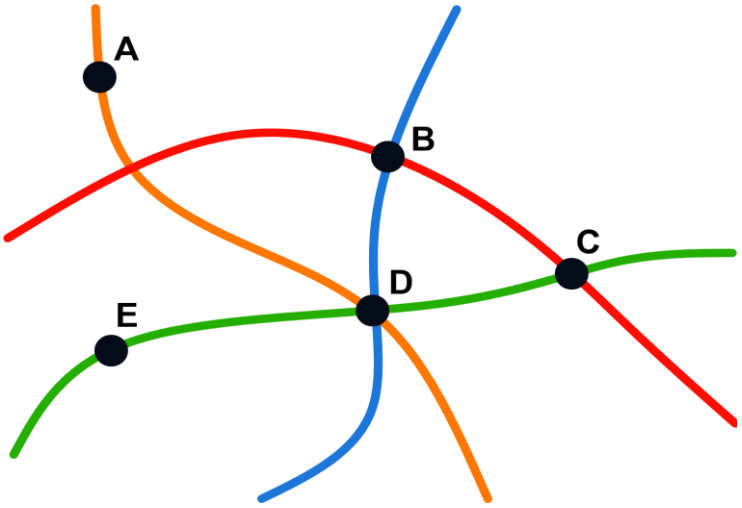
Tai informatika!

Informatikams tenka duomenyse identifikuoti struktūras (šablonus) ir jas panaudoti. Šiame uždavinyje naudojami struktūrizuoti tekstai (dainos), kuriami laikantis griežtos taisyklių sistemos. Tokie generavimo mechanizmai (diagramos) vadinami baigtiniais automatais ir naudojami projektuojant programavimo kalbas, t. y. kalbas, kurias gali „suprasti“ kompiuteris.

Dėsningumų (šablonų) atpažinimas yra vienas esminių elementų sprendžiant uždavinius. Atpažinus dėsningumą, panašias savybes, galima skaidyti uždavinį į mažesnes dalis ir konstruoti sprendimo kelią.

31. Miestai ir greitkeliai

Bebrai rado žemėlapį su 5 miestais ir 4 greitkeliais. Juodi taškai reiškia miestus, o spalvotos linijos žymi greitkelius.

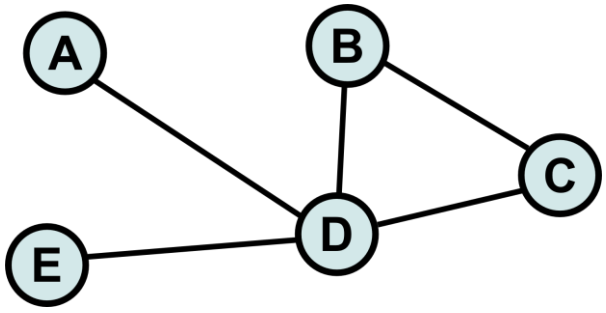


Kad geriau išsiaiškintų situaciją, bebrai nori pavaizduoti šį žemėlapį diagrama, kurioje:

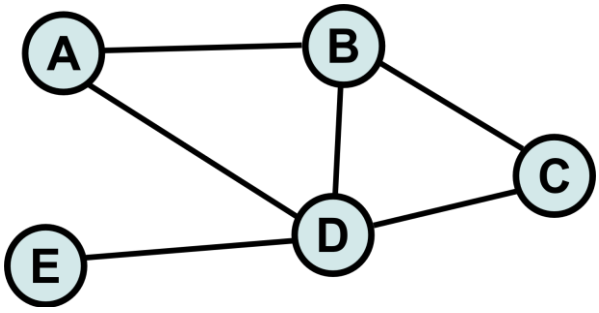
- miestai žymimi apskritimais;
- du prie tos pačios magistralės esantys miestai sujungiami atkarpa.

Kuri schema atitinka pradinį žemėlapį?

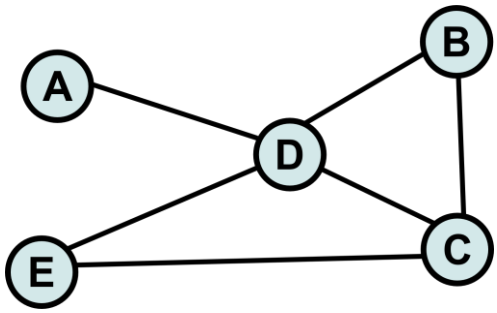
A.



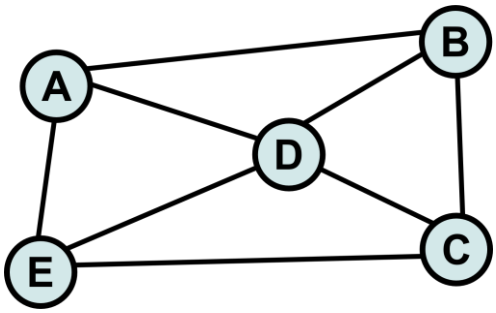
B.



C.



D.



Paaiškinimas

Atsakymas: C.

Atidžiai išnagrinėję žemėlapį galime pastebėti, kad:

- A miestas yra prie to paties greitkelio, kaip ir D miestas;
- B miestas yra prie to paties greitkelio, kaip C ir D miestai;
- C miestas yra prie to paties greitkelio, kaip B, D ir E miestai;
- D miestas yra prie to paties greitkelio, kaip ir visi kiti miestai.

Ir galiausiai, E miestas yra prie to paties greitkelio, kaip C ir D miestai.

Visi kiti atsakymai neteisingi.

- A atsakyme trūksta ryšio tarp miestų E ir C.
- D atsakyme yra ryšys tarp A ir B miestų, taip pat tarp A ir E, o tie miestai nėra prie tų pačių greitkelių.
- Tas pats pasakytina ir apie B atsakymą, kuriame taip pat nėra ryšio tarp E ir C miestų.

Kai kurie sprendėjai gali padaryti neatsargių klaidų. Svarbu pažymėti, kad:

- Nors iš A miesto galima pasiekti B miestą, jie iš tikrųjų nėra prie to paties greitkelio;
- Nors tarp E ir C miestų yra dar vienas miestas, jie vis tiek sujungti tuo pačiu greitkeliu.

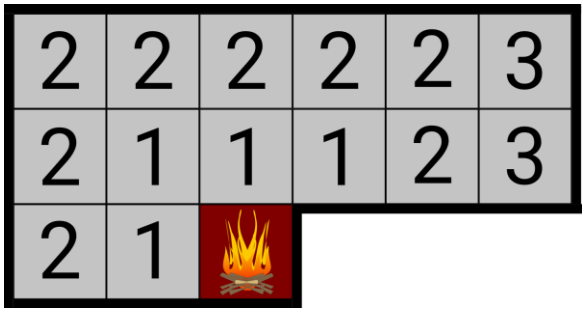
Tai informatika!

Kompiuterinėse programose tikrovė turi būti vaizduojama pateikiant kuriuos nors duomenis, kompiuteris juos apdoroja ir gauna rezultatus. Tą pačią realybę paprastai galima pavaizduoti keliais būdais, o kompiuterių programuotojai turi pasirinkti tą, kuris yra tinkamiausias nurodytai užduočiai (algoritmui) atlikti ir užrašyti programa. Todėl kompiuterių programuotojams būtina žinoti įvairius tų pačių duomenų vaizdavimo būdus.

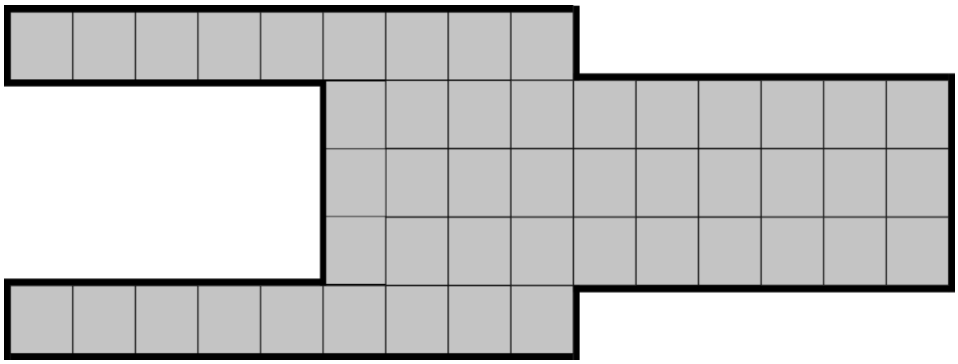
Vienas iš dažnai naudojamų vaizdavimo būdų yra grafas. Grafas sudarytas iš viršūnių (paprastai taškų ar apskritimų, nurodant pavadinimus juose arba šalia jų) ir briaunų (linijų, jungiančių viršūnes). Dalis informatikos (kur ji sutampa su diskrečiąja matematika) vadinama grafų teorija. Joje apibrėžiama daug algoritmų, naudingų darbui su duomenims, kuriuos galima vaizduoti grafais. Tačiau, kaip rodo ši užduotis, tą pačią realybę galima pavaizduoti kelių rūšių grafais – pirmasis paveikslėlis taip pat yra grafas (kai mes atsisakome greitkelių dalių, kurios nesujungia jokių dviejų miestų), tačiau jis sujungia miestus šiek tiek kitaip, nei sprendimo grafas.

32. Šildymas

Grupė bebrų nori apšildyti savo namą naudodami keturis šiltą orą pučiančius įrenginius. Namas sudarytas iš atskirų kambarių. Šildymo įrenginys užima vieną kambarį ir jį tučtuojau apšildo. Šiltam orui patekti ir apšildyti visus kaimyninius kambarius (kurie turi bendrą sieną ar kampą) prireikia 1 minutės. Pavyzdyje parodyta, kiek minučių prireiks visiems kambariams apšildyti, kai yra vienas šildymo įrenginys (kambarys su židiniu).



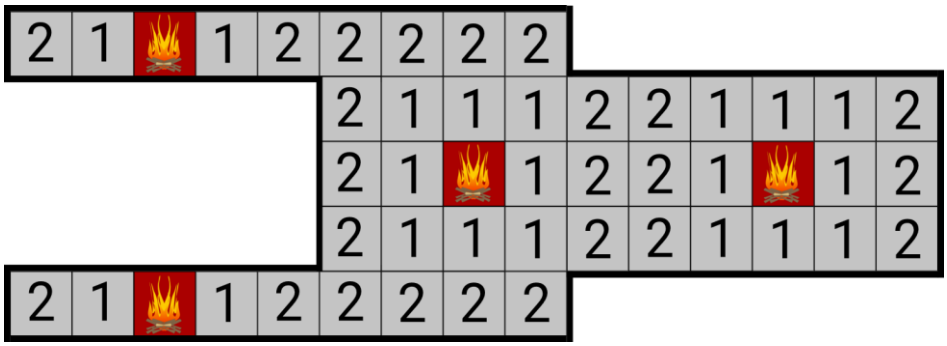
Išdėstyk 4 šildymo įrenginius taip, kad visas bebrų namas būtų apšildytas kiek galima greičiau. Spustelėk Kambarius, kuriuose turėtų būti šildymo įrenginiai.



Paaiškinimas

Teisingas atsakymas – 2 min.

Vienas iš galimų šildymo įrenginių išdėstymo būdų pateiktas paveiksle, jame sužymėta, per kiek minučių bus apšildytas kiekvienas kambarys. Matome, kad visas namas apšildomas per 2 minutes.

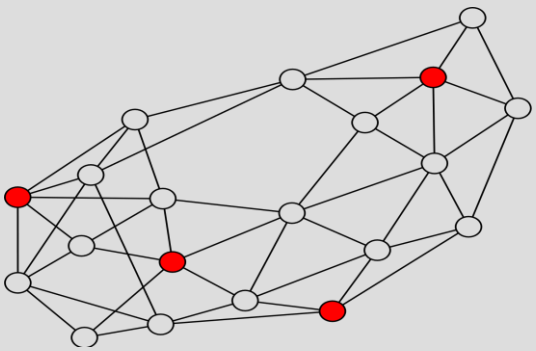


Apšildyti per 1 minutę neįmanoma. Kad namas būtų apšildomas per 1 minutę, visi kambariai turi būti kaimyniniai kuriam nors iš šildymo įrenginių. Name yra 48 kambariai . Kiekvienas šildymo įrenginys per 1 minutę daugiausia gali apšildyti 9 kambarius (įskaitant ir tą, kuriame jis yra įrenginys). Vadinasi, per vieną minutę galima būtų daugiausia apšildyti 36 kambarius.

Tai informatika!

Namo struktūra modeliuojama grafu. Kiekvienas kambarys vaizduojamas grafo viršūne. Dvi viršūnės jungiamos briauna (linija), jei kambariai yra kaimyniniai, t. y., turi bendrą sieną ar bendrą kampą kaip reikalaujama sąlygoje.

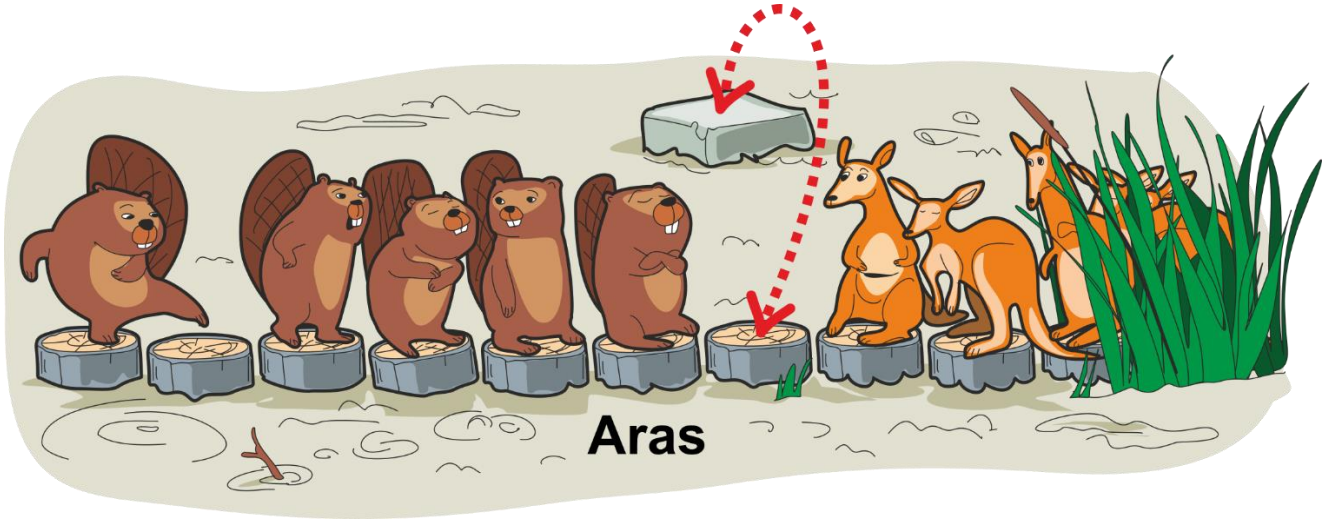
Turime rasti tokias keturias viršūnes grafe, kad visos viršūnės būtų pasiekiamos naudojant kuo mažiau briaunų. Šis uždavinys gali būti siejamas su paieška į plotį, bet dar geriau – su optimizavimo problema, vadinamąja dominuojančia seka, t. y., reikia rasti tokią minimalią viršūnių seką, iš kurių būtų pasiekiamos visos kitos viršūnės. Keturi šildymo įrenginiai ir yra šio uždavinio dominuojanti seka. Bendru atveju labai sunku rasti minimalią dominuojančią seką, tačiau yra metodų, kurie gerai tinka tam tikriems grafų atvejams. Pateikiame vieną tokį pavyzdį.



Tipiškas šio uždavinio pritaikymas – bevielio interneto taškų išdėstymas dideliame pastate. Viršūnės būtų pastato kambariai. Du kambariai kaimyniniai, jei jie yra ryšio zonoje. Kambariai, kurie sudarys minimalią dominuojančią seką, ir bus ieškomos vietos interneto taškams.

33. Bebrai ir kengūros

Eidami trinkelėmis grįstu taku per pelkę penki bebrai sutiko grupę kengūrų, einančių priešinga kryptimi. Niekas nenori įdribti į pelkę, tad turi likti ant trinkelių (ant vienos trinkelės telpa tik vienas gyvūnas). Kengūros išsiaiškino, kad nuo tam tikros trinkelės įmanoma nušokti ant netoliese esančio akmens ir vėl grįžti ant tos pačios trinkelės. Vėlgi, tik viena kengūra gali stovėti ant akmens vienu metu.



Kengūros ir bebrai, išskyrus bebrą Arą, neprieštarauja paeiti atgal tiek, kiek reikia. Tuo tarpu Aras iškėlė sąlygą, kad jis gali žengti atgal tik 10 žingsnių (vienas žingsnis - viena trinkelė).

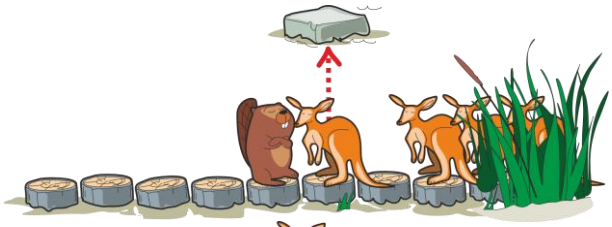
Atsižvelgdami į Aro reikalavimą, nustatykite, **kiek kengūrų galėtų prasilenkti?**

A. Daugiau nei 10 kengūrų gali prasilenkti su Aru.
B. Lygiai 10 kengūrų gali prasilenkti su Aru.
C. Lygiai 6 kengūros gali prasilenkti su Aru.
D. Lygiai 4 kengūros gali prasilenkti su Aru.
E. Mažiau nei 4 kengūros gali prasilenkti su Aru.
F. Neįmanoma nustatyti.

Paaiškinimas

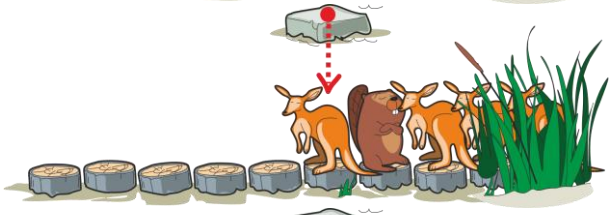
Teisingas atsakymas: lygiai 6 kengūros gali prasilenkti su Aru.
Bebras Aras yra svarbiausias, nes visi kiti bebrai sutinka eiti atgal tiek, kiek reikia. Taigi galima ignoruoti visus kitus bebrus ir nagrinėti tik Aro veiksmus.
Štai kaip viena kengūra prasilenktų su Aru:

Kengūra užšoka ant akmens:

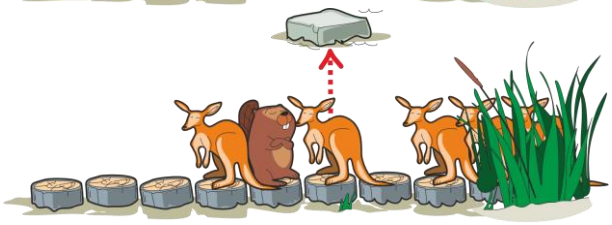


Aras paeina du žingsnius į priekį:

Kengūra grįžta atgal ant trinkelės ir žingsniuoja į priekį:



Aras atsitraukia du žingsnius atgal, kad leistų kitai kengūrai užšokti ant akmens:



Pakartojęs šiuos veiksmus 5 kartus Aras prasilenks su 5 kengūromis ir padarys 10 žingsnių atgal. Po to dar viena kengūra gali praeiti, nes Aras dar bus pasitraukęs nuo specialios trinkelės ir jam reikės eiti tik į priekį. Taigi iš viso 6 kengūros gali prasilenkti, kol Aras padarys 10 žingsnių atgal.

Tai galima išreikšti matematine formule. Jei Aras norėtų prasilenkti su k kengūrų, jis turėtų padaryti $s = 2 \times (k - 1)$ žingsnių atgal. Prasilenkiančių kengūrų skaičių galime apskaičiuoti šitaip: $k = 0.5 \times s + 1$.

Tai informatika!

Algoritmai yra kompiuterio pagrindas duomenims apdoroti ir veiksmams atlikti nusakyta tvarka.

Keičiamos kintamųjų reikšmės. Kiekviena trinkelė ir akmuo yra tarytum kintamieji, kur laikomos kintamos reikšmės, o bebrai ir kengūros yra tarytum duomenys, su kuriais dirbama.

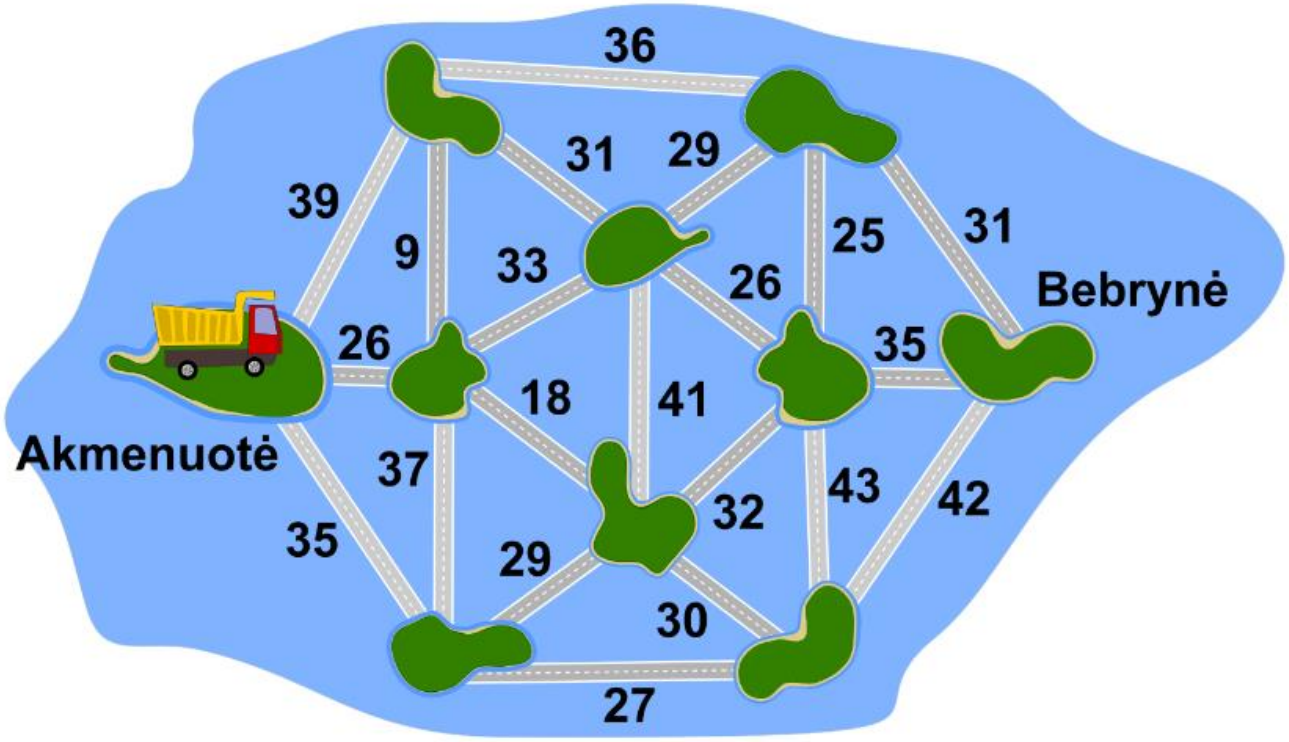
Žingsnių tvarka. Turi būti pateikta aiški tvarka, kaip judėti kengūroms, kad prasilenktų su bebrais, kai keliaujama priešingomis kryptimis.

Kartojami būtini veiksmai tiek kartų, kiek reikia. Šiuo atveju ta pati judėjimo seka yra kartojama keletą kartų. Tai tipiškas informatinio mąstymo uždavinys: išsprendi mažą pavyzdį ir pakartoji sprendimą, kiek reikia kartų.

Taigi algoritmas yra operacijų seka, įskaitant kartojimą, kuri gali būti modeliuojama kompiuterinėmis sistemomis. Atpažinus dėsningumus (šablonus) algoritmuose (kengūrų ir bebro kartojami žingsniai), juos galima užrašyti programa, o programą galima kartoti automatiškai daug kartų ir išspręsti pateiktą užduotį. Trinkelės ir akmuo yra tarytum kompiuterio procesoriaus registrai arba laikmena, galinti laikyti duomenis.

34. Bebrynės statybos

Bebrynės kraštą sudaro daugybė tiltais sujungtų salų. Ribojama kiekvienu tiltu važiuojančio transporto su kroviniu bendroji masė. Ribą viršijančios transporto priemonės negali važiuoti tuo tiltu.

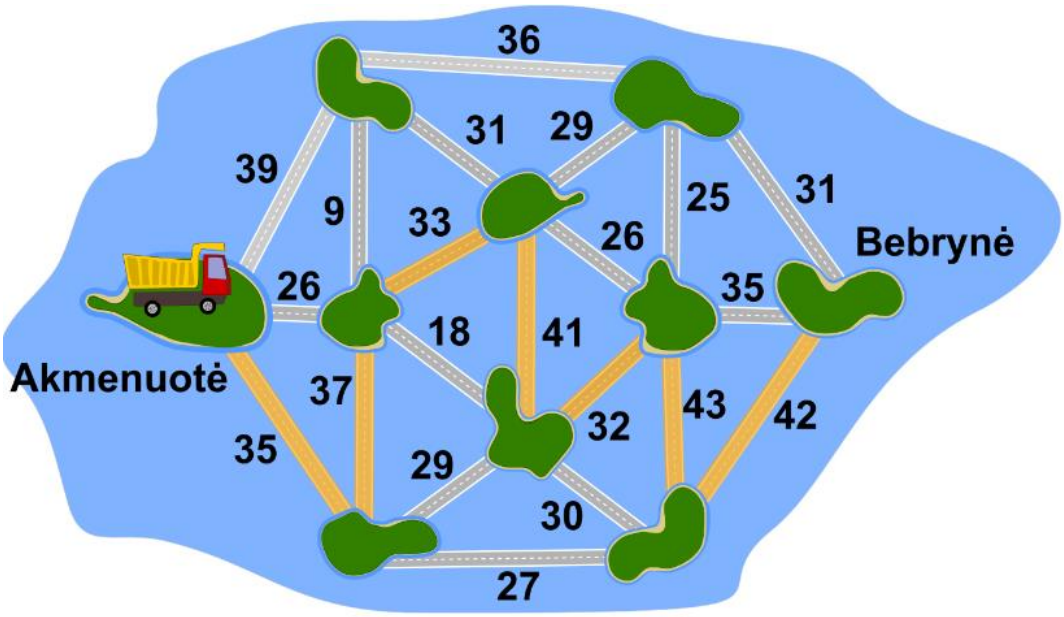


Bebrynės statybos įmonė turi pervežti smėlį iš Akmenuotės salos į Bebrynės salą.

Kokiu keliu turėtų važiuoti sunkvežimis, kad jis pervežtų didžiausią smėlio kiekį?

Paaiškinimas

Važiuojant iš Akmenuotės salos į Bebrynės salą, didžiausia sunkvežimio bendroji masė yra 32 tonos, kai pasirenkamas štai toks kelias:

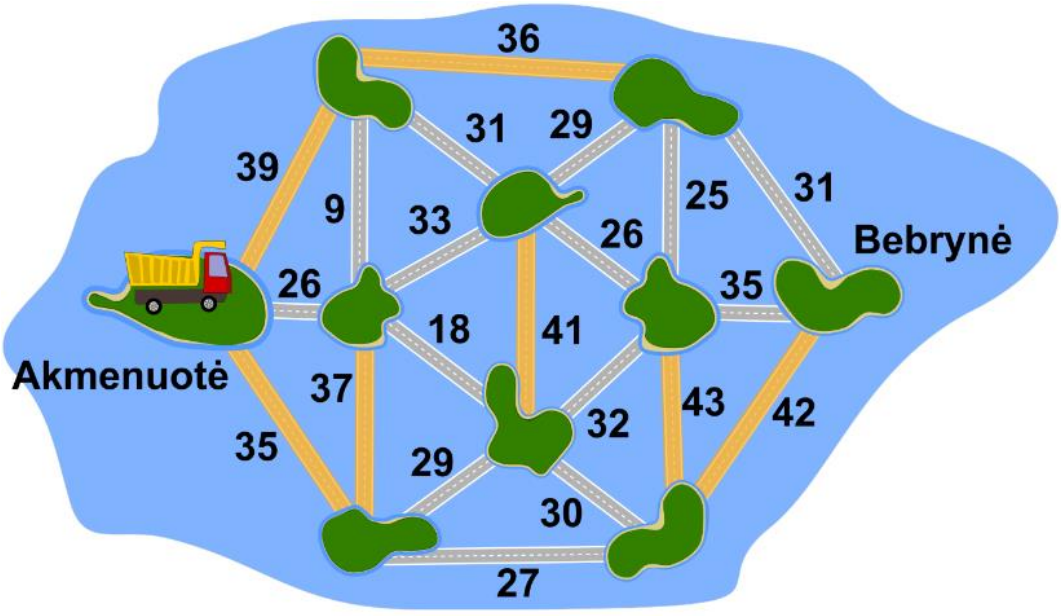


Norint rasti atsakymą, siūloma pradėti nuo galimai didesnio svorio, stebėti kokius tiltus galėtume naudoti, tada mažinti svorį, kol bus galima naudoti pakankamai tiltų Bebrynės salai pasiekti.

Galime pradėti nuo sunkiausio svorio, kurį gali išlaikyti bent vienas tiltas: 43 tonos. Tik vienas tiltas gali išlaikyti šį svorį, tad mes jį pasirenkame.

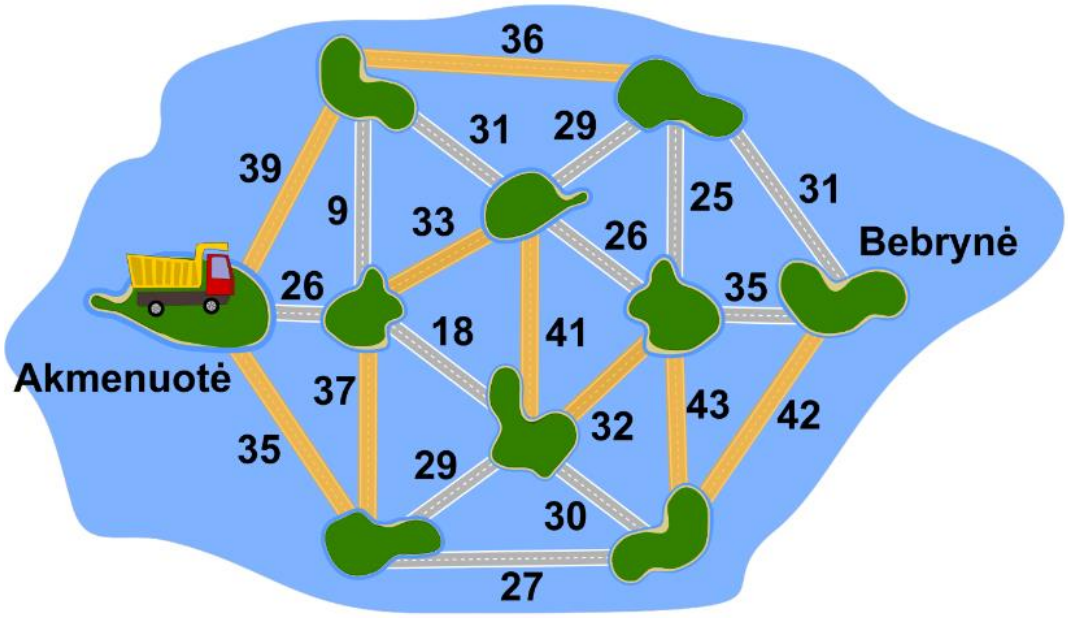
To nepakanka norint suformuoti kelią nuo Akmenuotės iki Bebrynės. Norėdami pervažiuoti daugiau tiltų, svorį sumažiname iki 42 tonų ir pridedame dar vieną tiltą.

To vis dar nepakanka, todėl toliau mažiname svorį, pridėdami tiltus, kurie gali išlaikyti 41, tada 39, 37, 36, tada 35 tonas.



Turėdami 35 tonų tiltą, jau pasirinkome 8 tiltus, tačiau kelio nuo Akmenuotės iki Bebrynės vis dar nėra.

Tęsiame su 33 tonų ribos tiltu, tada 32 tonų tiltu ir gauname šį galimų tiltų rinkinį:



Kai suformuojame užbaigtą kelią nuo Akmenuotės iki Bebrynės, galime panaikinti mums nereikalingus tiltus ir liktų tik teisingas kelias.

Tai informatika!

Pateiktas uždavinys yra grafų teorijos pralaidžiausio kelio ieškojimo pavyzdys. Jo taikymas apima maksimalaus pralaidumo tarp tinklo galinių taškų nustatymą ir kitas vadinamąsias siauriausios vietos problemas. Taip pat gali būti sprendžiama kaip dalinė problema, ieškant didžiausio pralaidumo kelio Fordo ir Fulkersono maksimalaus srauto algoritme.

Mūsų pasirinktas godusis algoritmas yra panašus į Kruskalio algoritmą, kuris naudojamas apskaičiuojant minimalų grafo apimties medį.

35. Dėlionė

Žaislų parduotuvėje bebriukas ant sienos mato paveikslą. Jis nori sudėlioti tokį patį paveikslą iš atskirų detalių, kurios yra pavaizduotos ant prekystalio sienelės. Ant kiekvienos detalės parašyta jos kaina monetomis. Bebriukas gali pirkti kokias nori ir kiek nori detalių ir jas bet kaip sukioti.

Kiek mažiausiai monetų bebriukui kainuoja visos reikalingos detalės?

A. 20 monetų

B. 16 monetų

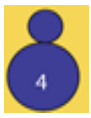
C. 13 monetų

D. 14 monetų

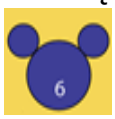
Paaiškinimas

Sprendžiant šį uždavinį svarbu pamatyti, kad detalė 

, kainuojanti tik vieną monetą, gali būti panaudota net 4 kartus. Pakeitus šią detalę kita 

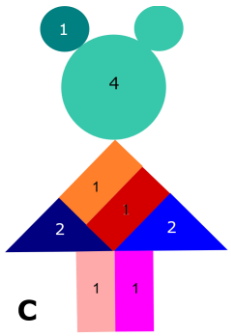
, kuri būtų panaudota tik 2 kartus, tačiau kainuotų net 10 monetų. Svarbu neapsirikti ir su dėlionės galva, nes detalę 

galima pasukti ir dar pridėti detalę 

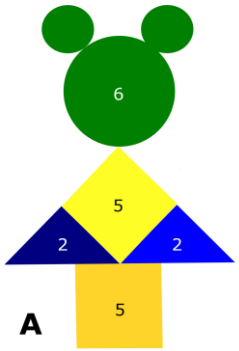
, kurios kaina tik 1 moneta. Detalė 

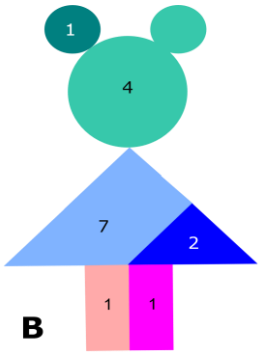
kainuoja 6 monetas, todėl pigiau naudoti 2 detales, kurios kainuoja 5 monetas. Telieka pridėti iš šonų po du trikampius 

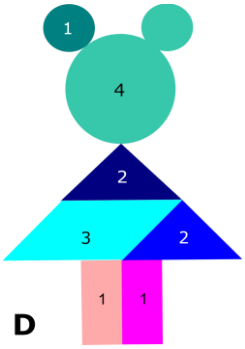
ir juos pasukti. Taigi visos dėlionės mažiausia kaina – 13 monetų.



Kiti paveikslėliai brangesni:





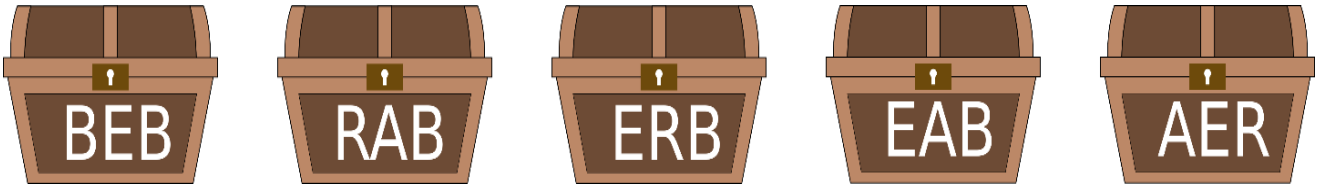


Tai informatika!

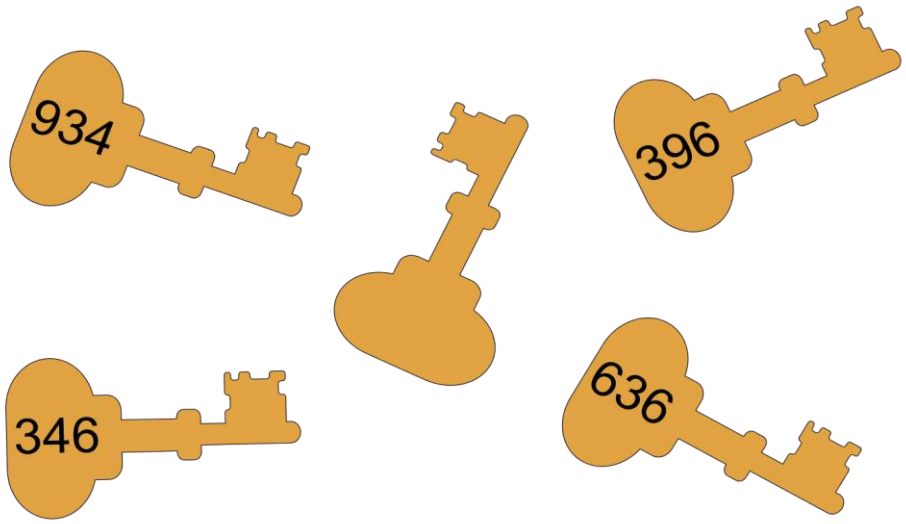
Sprendžiant šį uždavinį dėlionė suskaidoma į fragmentus ir tikrinama, kurios detalės geriausiai tinka fragmentams. Geriausio sprendimo kriterijus – visos dėlionės detalių mažiausia kaina, tai yra, mažiausios kainos radimas kiekvienam fragmentui. Programavime labai populiarūs algoritmai, kai ieškoma uždavinio geriausio sprendimo būdo pagal iš anksto nustatytą kriterijų.

36. Durininkas Jokūbas

Jokūbas dirba durininku name, kuriame yra penki butai,. Kiekviename bute gyvena po vieną bebrą. Išeidami į darbą bebrai palieka buto raktus Jokūbui. Raktus Jokūbas saugo po vieną atskirose dėžutėse. Kiekvienos dėžutės užrašas atitinka pirmąsias buto savininko vardo raides.



Rūpinantis saugumu ant raktų savininkų vardų užrašų nėra. Vietoje užrašų Jokūbas ant raktų priklijavo etiketes su trimis skaitmenimis, identifikuojančiais savininkus. Visiems raktams ta pati vardo raidė atitinka tą patį skaitmenį. Kartą visos dėžutės nukrito, raktai pabiro, o nuo vieno rakto etiketė nukrito ir pasimetė.



Kurio rakto etiketė pasimetė?

Paaiškinimas

Atsakymas: 496.

Dėžutė BEB yra vienintelė, kurios pirmoji ir trečioji raidės sutampa. Todėl šios dėžutės raktas yra 636. Tai reiškia, kad skaitmenį 6 atitinka raidė B, o skaitmenį 3 – raidė E. Vienintelė dėžutė AER baigiasi raide R, todėl skaitmuo 4 atitinka raidę R ir todėl šios dėžutės rakto etiketė yra 934. Tai reiškia, kad skaitmenį 9 atitinka raidė A. Atitinkamai dėžutę EAB atitinka raktas, kurio etiketė 396, o ERB – 346. Lieka dėžutė RAB, tad jos etiketė yra 496.

Tai informatika!

Šifravimas ir dešifravimas yra pagrindinės kriptografijos sąvokos. Šiame uždavinyje šifravimas ir dešifravimas remiasi unikalia ir nekintama šifravimo sistema, kuri vadinama Vatsyayana šifru. Tokio šifro idėja atėjo iš indiško teksto, parašyto IV amžiuje.

Šiame uždavinyje šifruojama tam tikrai raidei priskiriant tam tikrą skaitmenį. Raidė ir skaitmuo sudaro unikalią porą ir raidės ar skaitmens pasikartojimai draudžiami. Šifruojant pranešimą kiekviena raidė pakeičiama ją atitinkančiu skaitmeniu, o dešifruojant – skaitmuo keičiamas raide. Šis būdas yra nesaugus, nes visame pranešime naudojama nesikeičianti raidės ir skaitmens atitiktis. Sužinojus tokią atitiktį, galima skaityti pranešimą toliau, nes atitiktis nesikeičia.

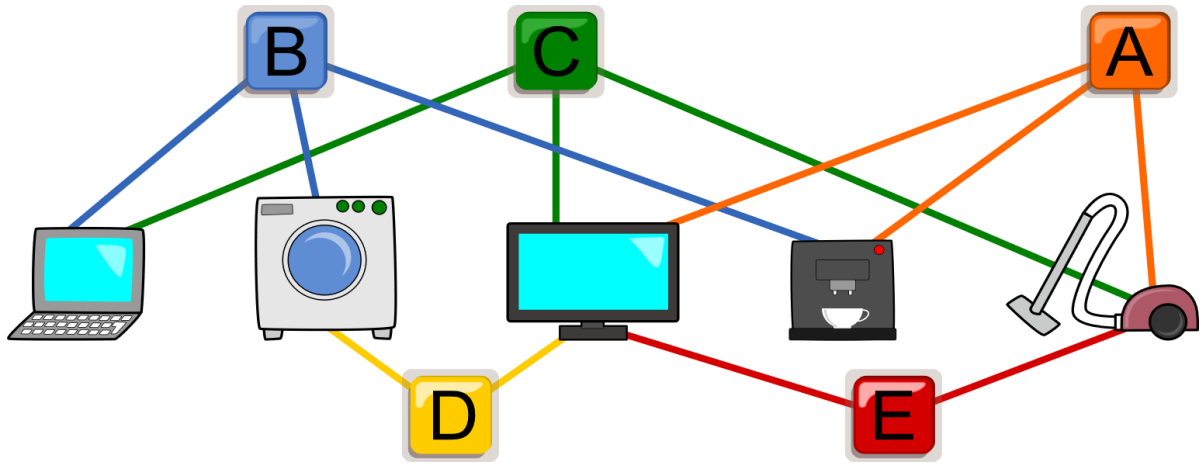
37. Buitinės technikos valdymas

Bebriuko Broniaus namuose yra penki buitiniai prietaisai ir penki mygtukai (žr. paveikslą) tiems prietaisams valdyti. Mygtukais galima įjungti arba išjungti prietaisą. Tačiau yra ir nepatogumų, kadangi kiekvienu mygtuku valdomi keli prietaisai:

- Mygtukas A yra prijungtas prie televizoriaus, kavos aparato ir dulkių siurblio;
- Mygtukas B prijungtas prie kompiuterio, skalbimo mašinos ir kavos aparato;
- Mygtukas C prijungtas prie kompiuterio, televizoriaus ir dulkių siurblio;
- Mygtukas D yra prijungtas prie skalbimo mašinos ir televizoriaus;
- Mygtukas E yra prijungtas prie televizoriaus ir dulkių siurblio.

Spausdami mygtukus įjunkite tik televizorių ir kavos aparatą!

Paspauskite tik vieną mygtuką vienu metu ir stebėkite, kurie prietaisai veikia.



Paaiškinimas

Mygtukų seka, kuriuos reikia paspausti, kad įjungtumėte tik televizorių ir kavos aparatą yra:

B + D + C + E

Norėdami rasti mygtukų derinius, kurie lemia tam tikrą prietaisų būseną, galime pastebėti, kad tam tikri mygtukų deriniai gali įjungti tik vieną prietaisą. Pirmiausia galime pastebėti paprastus derinius:

- A + E įjungia tik kavos aparatą
- C + E įjungia tik kompiuterį

Tada galime pastebėti, kad skalbimo mašina valdoma paspaudžiant B, tada išjungiant kompiuterį ir kavos aparatą, kuriuos abu mes dabar žinome, kaip valdyti. Taigi skalbimo mašiną valdo B + A + E + C + E = A + B + C (nes E + E panaikina vienas kito veiksmus). Tokiu būdu galime gauti visą derinių, skirtų valdyti kiekvieną įrenginį, sąrašą:

- Kompiuteris: C + E
- Kavos aparatas: A + E
- Skalbimo mašina: A + B + C
- Televizorius: A + B + C + D
- Dulkių siurblys: A + B + C + D + E

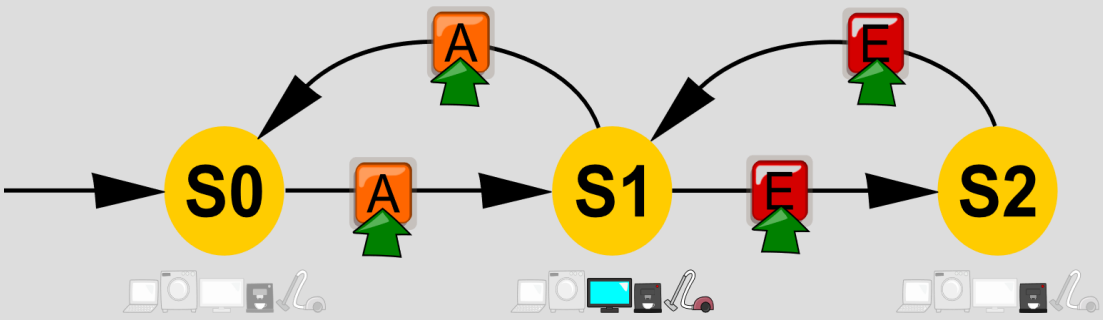
Taigi, norėdami įjungti televizorių ir kavos aparatą, turime paspausti atitinkamus derinius:
A + B + C + D + A + E arba tiesiog B + C + D + E, nes A + A panaikina vienas kito veiksmus.

Tai informatika!

Prietaisų ir mygtukų sistema bei įjungimo bei išjungimo būsenų pokyčiai gali būti modeliuojami vadinamuoju baigtiniu automatu. Baigtiniai automatai naudojami kompiuterių moksle sistemoms modeliuoti, vaizduojant visas įmanomas būsenas ir visus perėjimus iš vienos būsenos į kitą.

Šiai užduočiai atlikti galime naudoti baigtinį automatą, kad pavaizduotume visas įmanomas įjungimo ir išjungimo būsenas elektrinių prietaisų rinkinyje ir perėjimus, kuriuos sukelia skirtingų mygtukų paspaudimas.

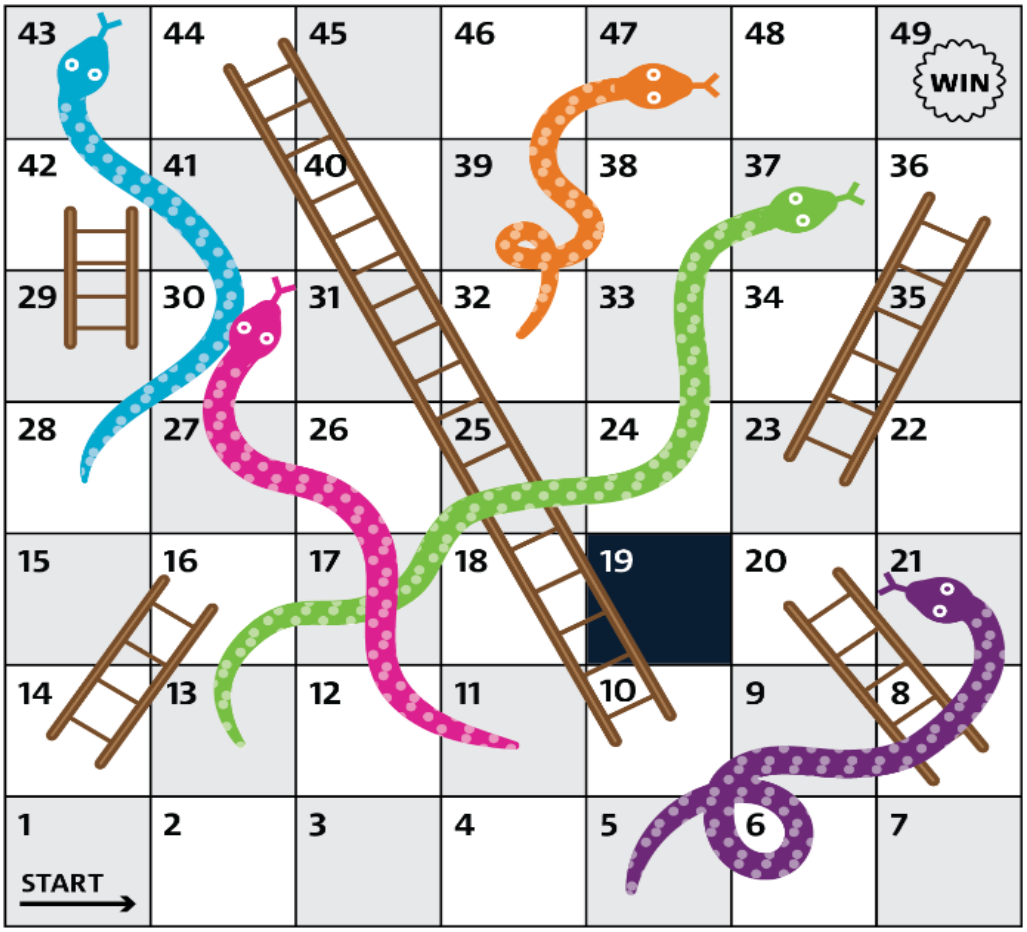
Mūsų užduotis atitinkančiame baigtiniame automate yra pradinė būsena (S0), kai visi prietaisai išjungti. Tada, paspaudus mygtuką A, pereiname į būseną S1, kurioje kompiuteris ir skalbimo mašina yra išjungti, o televizorius, kavos aparatas ir dulkių siurblys įjungti. Iš S1, jei dar kartą paspaudžiame A, grįžtame į S0, kai visi prietaisai išjungti. Tačiau jei paspaudžiame, pavyzdžiui, mygtuką E, pereiname į būseną S2, kurioje visi prietaisai, išskyrus kavos aparatą, yra išjungti, kaip parodyta paveiksle. Taigi gavome kavos virimo aparato valdymo metodą: reikia naudodotis mygtukais A + E, kaip paaiškinta atsakymų skyriuje.



Panašiai, plėtodami visus baigtinio automato perėjimus, galėtume rasti derinius, kurie valdo kiekvieną prietaisą.

38. Kopėčios ir gyvatės

Žaidžiant kopėčių ir gyvačių žaidimą metamas kauliukas (taškai nuo 1 iki 6) ir paeinama tiek langelių, kiek taškų atsivertė. Žaidėjai pradeda nuo 1-ojo langelio ir laimi, kai pasiekia paskutinį langelį (49).



Jei patenkate ant gyvatės galvos, nuslystate žemyn iki uodegos. Pavyzdžiui, jei atsistosite ant 21-ojo langelio, tai gyvatė numes jus į 5-ąjį langelį. Jei atsistojate ant langelio ties kopėčių apačia, tai lipate iki kopėčių viršaus. Pavyzdžiui, jei atsistosite ant 23-iojo langelio, tai užlipsite į 36-ąjį langelį.

Jei stovite ant 19-ojo langelio, kiek mažiausiai kartų teks mesti kauliuką, kad laimėtumėte?

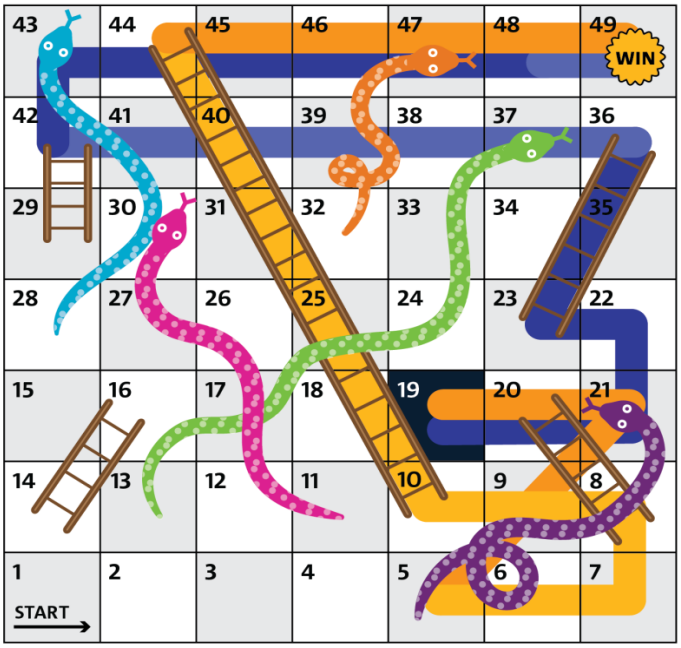
A) 2

B) 3

C) 4

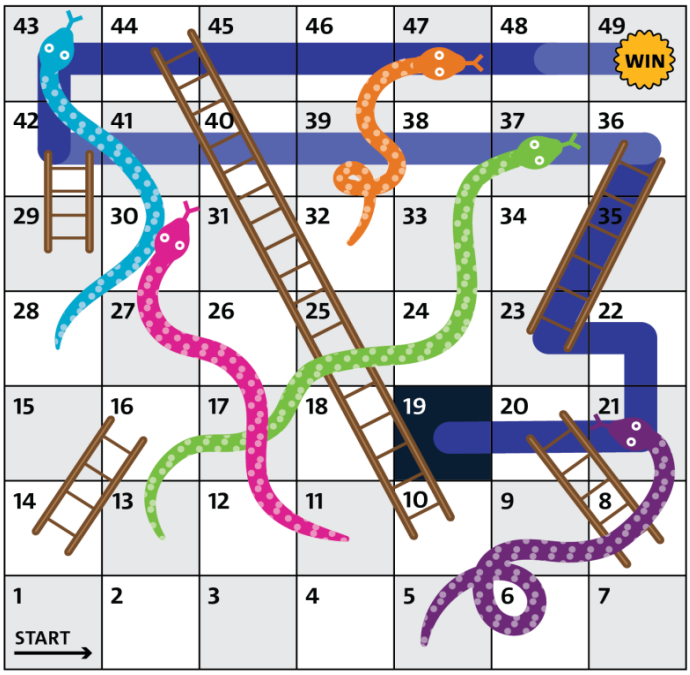
D) 5

Paaiškinimas

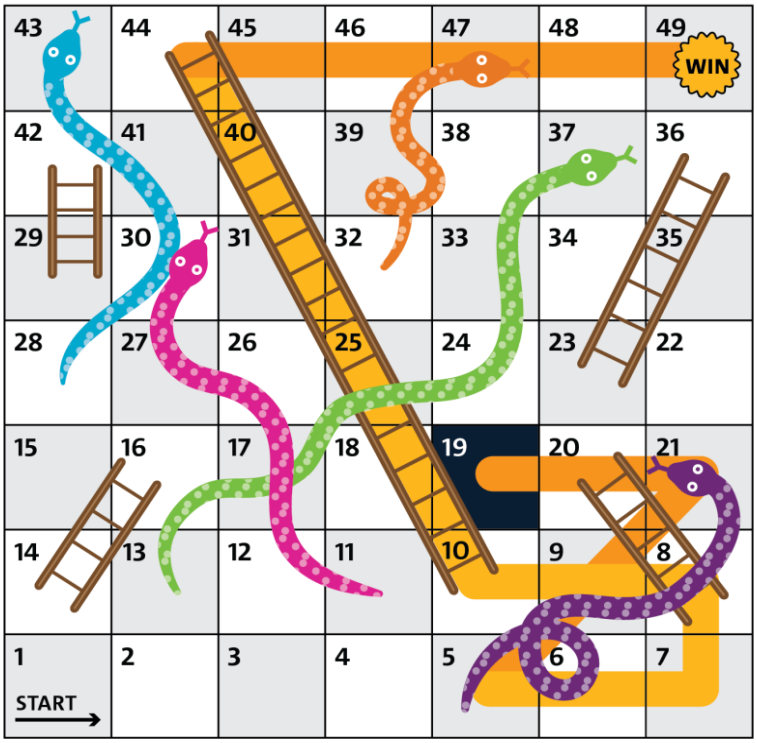


Teisingas atsakymas: 3.

Einant tik į priekį trumpiausiu keliu prireikia 4 kauliuko metimų. Iš 19-ojo langelio, išmetus 4, pasiekiamas 23 langelis ir kopėčios užkelią į 36-ąjį langelį. Nuo čia daugiau kopėčių nėra, lieka 13 langelių, taigi reikia 3 metimų (6, 6, 1). Iš viso būtų 4 metimai.



Tačiau yra trumpesnis kelias – tam reikia paeiti atgal ir pasinaudoti ilgiausiomis kopėčiomis iš 10-ojo langelio į 44-ąjį. Iš 19-ojo langelio, jei išmetate du, pereinate į 21-ąjį langelį ir gyvatė numeta jus į 5-ąjį langelį. Iš šio langelio, jei išmesite penkis, pateksite į 10-ąjį langelį, iš kurio kopėčiomis užkopsite į 44-ąjį langelį. Čia, išmetę penkis, pateksite į laimimą langelį – iš viso prireikė 3 metimų.



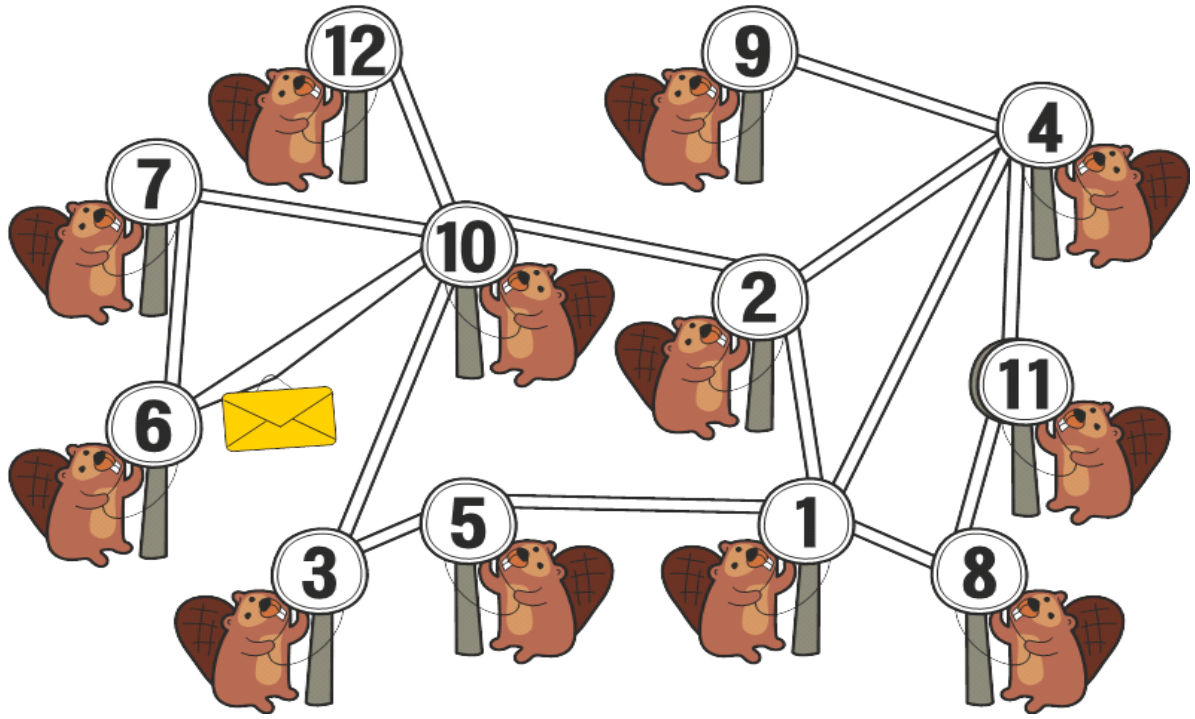
Neįmanoma laimėti 2 metimais, kadangi pirmu metimu galima patekti į 20, 5, 22, 36, 24 arba 25 langelius (metant kauliuką), tačiau toliau vieno metimo neužtenka, norint pasiekti galutinį 49-ąjį langelį.

Tai informatika!

Tai trumpiausiojo kelio uždavinys. Trumpiausio kelio problema paprastai vaizduojamu grafu. Grafas yra abstrakti duomenų struktūra, kurioje viršūnės (langeliai šio žaidimo atveju) jungiamos briaunomis (linijomis). Sutarta, kad langeliai jungiami briauna, jei iš vieno jų galima pasiekti kitą vienu kauliuko metimu. Šitaip pavaizdavus žaidimo lentą, uždavinys – rasti kelią, kurį sudarytų mažiausias briaunų skaičius ir kuris vestų į finišo langelį. Paieškos į plotį algoritmas (angl. *breadth-first search*) yra taikomas paieškai grafų struktūrose. Pradedama nuo pradinės viršūnės (ar vienos iš viršūnių) ir tikrinamos visos kaimyninės šio lygio viršūnės; patikrinus imamasi kito lygio viršūnių. Taigi šis algoritmas mūsų žaidimo atveju būtų pradedamas nuo 19-ojo langelio, išnagrinėjami visi galimi kauliuko metimai ir tik tada pereinama į kitą langelį.

39. Naujienų plitimas

Tvenkinyje gyvena 12-os bebrų kolonija. Kiekvienas bebras turi atskirą urvą. Tarp kaimynų ar šiaip geresnių draugų yra nutiestos virves, kuriomis vieni kitiems perduoda naujienas.



Bebrai nori, kad naujiena kuo greičiau pasiektų visus kolonijos gyventojus. Todėl kiekvienas gautą naujieną visomis turimomis virvėmis vienu metu persiunčia kitiems. Pavyzdžiui, jeigu naujieną sužinojo 8-ojo urvo gyventojas, tai ją persiunčia 1 ir 11 urvo gyventojams, po to šie persiunčia į 2, 4, 5 urvus ir t. t., kol visi bebrai sužino naujieną.

Kuriam bebrui reikia pasakyti naujieną, kad ji kuo greičiau pasiektų visus bebrus?

Paaiškinimas

Atsakymas: 2.

Jeigu naujiena būtų pasakyta 2-ajam bebrui, tai visus kitus bebrus ji pasiektų per 2 žingsnius. Pirmuoju žingsniu informuojami 4, 1 ir 10 bebras, o antruoju – visi likusieji.

Tai informatika!

Daugelį praktinių uždavinių galima išspręsti panaudojant grafų teoriją. Grafas – tai rinkinys taškų, vadinamų viršūnėmis (arba mazgais), ir viršūnes jungiančių linijų, vadinamų briaunomis. Šiame uždavinyje ieškota viršūnės, vadinamos grafo centru (arba Jordano centru pagal jo atradėją Camille Jordan, 1838–1922). Grafas gali turėti kelis centrus.

Vienas iš grafo centro panaudojimų – jame tikslinga pastatyti visas grafo viršūnes aptarnaujančią įmonę ar įstaigą. Tada bus mažiausias atstumas iki labiausiai nutolusios viršūnės. Kai grafas nedidelis (kaip šiame uždavinyje), grafo centrą galima rasti išbandžius visus variantas. Kai grafas didesnis, tenka pasitelkti kompiuterį. Šiam tikslui yra sukurtas Floido ir Varšalio algoritmas.

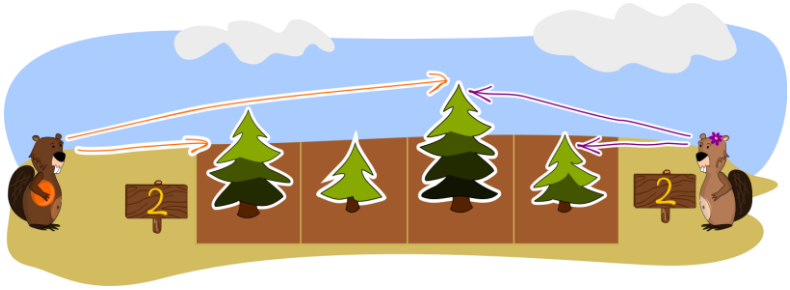
40. Medžių sudoku 4x4

Bebrų laukas padalintas į 16 plotelių, kurie išdėstyti tinkleliu 4x4. Į kiekvieną plotelį sodinama po vieną medį. Medžiai yra trijų aukščių: 1(), 2(), 3() ir 4(), po keturis kiekvieno aukščio.

Bebrai sodina medžius pagal šią taisyklę:

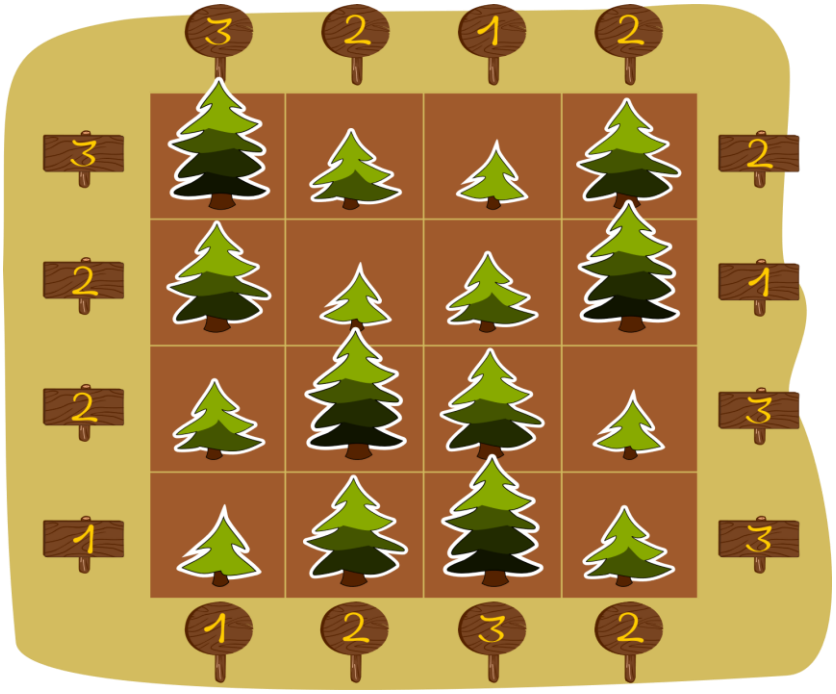
- kiekvienoje eilutėje (horizontaliai) turi būti tik vienas to paties aukščio medis,
- kiekviename stulpelyje (vertikaliai) turi būti tik vienas to paties aukščio medis.

Jei bebrai žiūri į medžius viena linija, tai jie nemato medžių, kuriuos užstoja aukštesni medžiai (žr. paveikslą).



Aplink lauką ties kiekviena eilute ir stulpeliu pastatyti skydeliai, kuriuose užrašyta, kiek matoma medžių. Bebriukas perpiešė lauką popieriaus lape, skydelių skaičius surašė teisingai, deja, kai kurių medžių aukštis neteisingas.

Raskite langelius, kur jis padarė klaidas, ir pataisykite klaidingų medžių aukščius. Spustelėkite pele medį, norėdami keisti jo aukštį.



Paaiškinimas

Pažvelgę į bebriuko sudarytą lentelę, matome, kad ji taisyklinga: kiekvienoje eilutėje ir kiekviename stulpelyje yra po vieną to paties aukščio medį. Tačiau numeriai skydeliuose neatitinka realaus vaizdo.

Pirmiausia reikia rasti teisingas eilutes ir stulpelius. Atidžiai palyginę kiekvieno skydelio skaičius su realiai matomais medžiais, nustatome, kad 2 ir 3 eilutės ir 2 ir 4 stulpeliai yra teisingi ir jų nereikia keisti.

Kitose eilutėse ir stulpeliuose, kurių skydelių skaičiai neatitinka matomų medžių, kai kurių medžių aukščiai turės būti pakeisti. 1 ir 4 eilučių ir 1 ir 3 stulpelių skydelių skaičiai neatitinka realiai matomų medžių, vadinasi, reikia keisti kai kurių medžių aukščius.

Logiškai sumąstome, kad užtektų pakeisti šių eilučių ir stulpelių sankirtų langeliuose esančių medžių aukščius (paraudoninti langeliai), šios eilutės ir stulpeliai taptų teisingais, o buvusios teisingos eilutės ir stulpeliai nepakistų. Pirmame stulpelyje pakaktų sukeisti raudonai pažymėtus langelius. Tas pats su 3-iu stulpeliu. Taigi pakeitę medžių aukščius šiuose keturiuose langeliuose, išspręsime uždavinį.

Tai informatika!

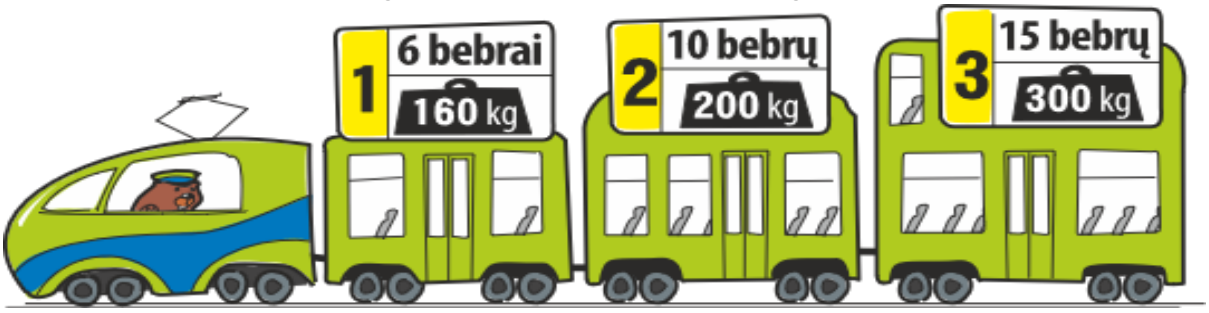
Šiuo uždaviniu ugdomos trys esminės informatikos kompetencijos. Pirmoji – tai gebėjimas rasti sprendimą pagal duotus ribojimus arba tenkinant nurodytas sąlygas. Antroji kompetencija – gebėjimas rekonstruoti objektą iš dalinės informacijos panaudojus duotas objekto savybes. Visa tai gali būti naudojama suglaudintiems objektams vaizduoti. Konceptas „glaudinimas neprarandant“ informatikoje vartojamas aprašyti procesui, kai glaudinant duomenis vartotojas nepraranda jokios informacijos, kuri buvo esant pradiniam pavidalui. Trečioji kompetencija susijusi su programavimu, tiksliau, testavimu ir verifikavimu, klaidų taisymu. Klaidų taisymas yra bendras informatikos metodas ir užtikrina duomenų judėjimo patikimumą, kad neprarastume, nesugadintume duomenų juos perduodami įvairiais kanalais. Skydeliuose nurodomi skaičiai yra pavyzdys, kaip galima automatiškai aptikti klaidas.

41. Kelionė traukiniu

Aštuonios bebrų šeimos nori keliauti traukiniu. Traukinyje yra tam tikras vietų skaičius, o ir bagažo svoris ribojamas. Šeimų duomenys pateikti lentelėje:

Šeima	*Narių skaičius*	*Bagažo svoris, kg*
Antanų (A)	3	50
Beržų (B)	4	80
Chemikų (C)	5	110
Danielių (D)	4	80
Elnių (E)	2	40
Filosofų (F)	3	70
Genių (G)	6	130
Herojų (H)	5	100

Geležinkelių kompanijos vagonuose turimų vietų ir bagažo ribojimų reikalavimai pateikti paveiksle:



Kiek šeimų gali keliauti, jei:

- vienoje vietoje gali sėdėti tik vienas bebras,
- visi tos pačios šeimos nariai turi keliauti tuo pačiu vagonu,
- šeimos bagažas turi būti padėtas tame pačiame vagone, kuriame sėdi šeima, ir
- bagažo svoris turi tenkinti vagono bagažo svorio ribojimus?

Paaiškinimas

Teisingas atsakymas: 7.

Visose šeimose iš viso yra 32 bebrai, tačiau traukinyje tik 31 vieta. Tai reiškia, kad visos 8 šeimos keliauti traukiniu negalės.

Viena iš galimybių susodinti 7 šeimas traukinyje, tenkinant esamus ribojimus, yra tokia:

1 vagonas: G šeima – 6 bebrai, 130 kg

2 vagonas: A, C, E šeimos – 10 bebrų, 200 kg

3 vagonas: B, D, H šeimos – 13 bebrų, 260 kg

Tai informatika!

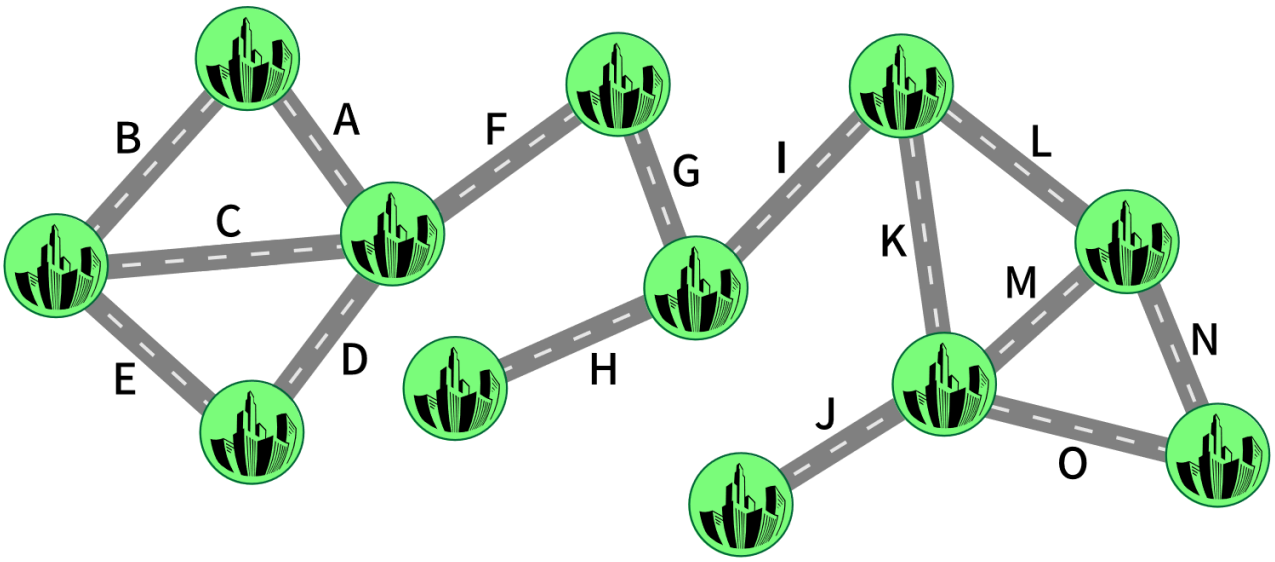
Yra žinomas šeimų skaičius, kiekvienos šeimos narių skaičius ir kiekvienos šeimos bagažo svoris. Tačiau pateikti ribojimai ne tik vietų skaičiui iš viso, bet kiekvienam traukinio vagonui. Be to, vagonuose ribojamas bagažo svoris, o tos pačios šeimos nariai negali keliauti skirtinguose vagonuose.

Tinkamas sprendinys – bet kuris sprendinys, tenkinantis apibrėžtus ribojimus. Geriausias iš tinkamų sprendinių – tai optimalus sprendinys, kuris ne tik tenkina visus ribojimus, bet ir leidžia keliauti didžiausiam skaičiui šeimų.

Šis uždavinys susijęs su kuprinės uždaviniu – gerai žinomu kombinatorinės optimizacijos uždaviniu, kuriam spręsti taikomi informatikos algoritmai.

42. Epidemija

12 Bebrijos miestų sujungti greitkeliais, kaip parodyta paveiksle. Miestai įvardyti raidėmis nuo A iki O.



Miestai, sujungti tiesiogiai arba netiesiogiai per kitus miestus vienu ar daugiau greitkelių, sudaro ekonominę bendriją. Visi 12 Bendrijos miestų priklauso tai pačiai bendrijai.

Plintant epidemijai, miestų merai nutarė apriboti eismą tarp miestų ir uždaryti du greitkelius (įrengiant užtvaras). Jie siekia padalinti Bebriją į tris atskiras ekonomines bendrijas. Kadangi norima minimizuoti ekonominę atskirtį užblokavus greitkelius, sutarta, jog mažiausia iš trijų bendrijų turi turėti kiek galima daugiau miestų.

Kuriuos du greitkelius reikia uždaryti?

Paaiškinimas

Teisingas atsakymas: F ir I – užblokavus šiuos greitkelius kiekvienoje iš trijų bendrijų bus 3, 4 ir 5 miestai.

Pirmiausia pastebėkime, kad galima ignoruoti kai kuriuos greitkelius, kadangi juos užtvėrus ekonominė bendrija nesikeičia. Pavyzdžiui, jei uždarysime C kelią, bus galima keliauti aplinkkeliais naudojantis A ir B keliais. Atidžiau panagrinėję matome, kad tik uždarę vieną iš penkių greitkelių F, G, H, I ir J, galime padalinti bendriją į smulkesnes dalis. (Reikia pripažinti, kad uždarę abu greitkelius A ir B vienu metu gautume vieną atsiskyrusį miestą ir jau būtume uždarę du greitkelius, taigi gautume tik dvi atskiras bendrijas vietoj trijų.)

Dabar bandykime uždaryti po du greitkelius iš minimų penkių ir kiekvieną kartą tikrinkime mažiausios bendrijos dydį. Pavyzdžiui, jei uždarysime F ir G greitkelius, tai gausime tris bendrijas, kurios turės atitinkamai 4, 1 ir 7 miestus, taigi mažiausią bendriją sudarys 1 miestas.

Kitos dviejų greitkelių kombinacijos, kurias taikant gaunama mažiausia bendrija iš 1 miesto, yra FH, FJ, GH, GJ, HI, HJ ir IJ.

Uždarę G ir I kelius gautume tris bendrijas, kurių dydžiai 5, 2 ir 5, o mažiausia bendrija yra iš 2 miestų.

Liko tik viena kombinacija – FI. Šiuo atveju bendrijų dydžiai yra 3, 4 ir 5, o mažiausia bendrija yra iš 3 miestų.

Kadangi reikalaujama, jog mažiausia bendrija turėtų kiek galima daugiau miestų, pastaroji kombinacija ir yra tai, ko mes ieškome.

Tai informatika!

Šio uždavinio esmė – rasti visus sujungtus komponentus neorientuotame grafe, kuriame kiekviena viršūnių pora yra sujungta bent vienu keliu. Norėdami rasti visus jungiančius komponentus, turime nagrinėti grafą: pradedame nuo pradžios ir sistemiškai tikriname visas viršūnes, sujungtas su šia viršūne tol, kol nebepasieksime naujų viršūnių. Šiame uždavinyje taip pat ieškoma tiltų. Tiltas yra briauna, kurią pašalinus padidėja nesujungtų grafo komponentų. Sujungtų komponentų analizė atliekama vaizdų apdorojimo programose norint rasti tam tikras vietas. Pavyzdžiui, žymenų segmentacijai, sluoksnių atskyrimui apdorojant vaizdus medicinoje ar vaizdų stebėjimui.

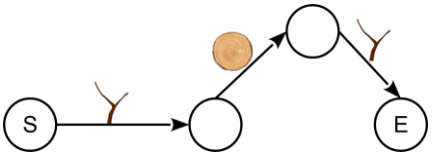
43. Slaptažodžiai

Kad apsaugotų savo buveinę, bebrai naudoja slaptažodžius, sudarytus tik iš dviejų simbolių:  ir  .

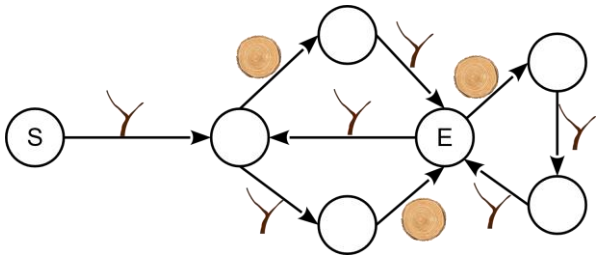
Ar slaptažodis yra tinkamas, parodo slaptažodžių tikrintuvas. Norėdami paaiškinti, kaip veikia jų slaptažodžių tikrintuvas, bebrai naudoja schemas iš rodyklių ir apskritimų. Tikrintuvas veikia pagal tokias taisykles:

- Slaptažodžių tikrintuvas visuomet pradeda darbą apskritime S;
- Tikrintuvas skaito slaptažodį po simbolį iš kairės į dešinę. Kiekviename apskritime tikrintuvas perskaito vieną slaptažodžio simbolį;
- Jeigu iš dabartinio apskritimo išeina rodyklė su perskaitytu slaptažodžio simboliu, tikrintuvas rodyklės kryptimi pereina prie kito apskritimo; priešingu atveju slaptažodžių tikrintuvas nutraukia darbą ir slaptažodžio nepriima;
- Jeigu visi simboliai perskaityti, tikrintuvas sustoja. Slaptažodis yra priimamas tik tuo atveju, kai tikrintuvas sustoja apskritime E.

Šiame pavyzdyje slaptažodžių tikrintuvas priima vienintelį slaptažodį  :



Bebrai patobulino slaptažodžių tikrintuvą, kaip parodyta schemoje:



Kuriuos iš pateiktų slaptažodžių naujasis slaptažodžių tikrintuvas priima?

A. 

B. 

C. 

D. 

E. 

Paaiškinimas

A ir B atsakymo variantai teisingi. Abiem atvejais slaptažodžių tikrintuvas perskaitęs paskutinį simbolį sustoja apskritime E.

Variantas C neteisingas: pirmasis simbolis yra  , tačiau visi tinkami slaptažodžiai turi prasidėti  .

Variantas D neteisingas: jis sudarytas iš 10 simbolių, tačiau visų tinkamų slaptažodžių ilgis dalijasi iš 3. Jeigu patikrinsite slaptažodį su slaptažodžių tikrintuvu, įsitikinsite, kad jis sustoja prieš devintąjį simbolį.

Variantas E neteisingas:  simbolis pasikartoja 5 kartus, o  simbolis pasikartoja 4 kartus. Tačiau tinkamuose slaptažodžiuose  pasikartoja dvigubai dažniau, nei  . Jei patikrintumėte slaptažodį su tikrintuvu, pamatytumėte, kad tikrintuvas sustoja prieš paskutinį simbolį.

Tai informatika!

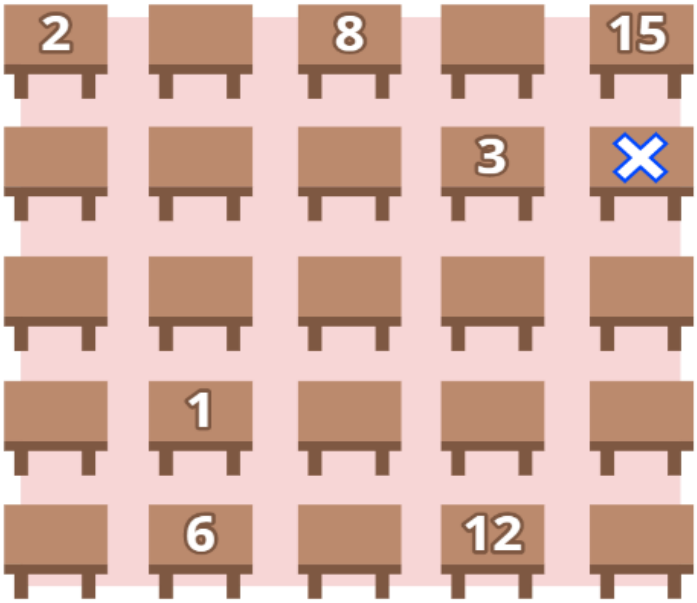
Slaptažodžių tikrintuvai, pademonstruoti šioje užduotyje, yra modeliuojami kaip baigtiniai auto-matai (angl. *finite-state machine*). Tai yra abstrakčios mašinos, galinčios vienu metu būti vienoje iš baigtinio skaičiaus būsenų. Baigtinis automatas gali pakeisti savo būseną į kitą, reaguodamas į įvestį. Baigtinis automatas apibrėžiamas savo būsenų sąrašu, pradine būsena ir taisyklėmis, pagal kurias, priklausomai nuo įvesties, baigtinis automatas keičia būsenas.

Baigtinių automatų veikimą galime atpažinti daugybėje šiuolaikinės visuomenės įrenginių, at-liekančių iš anksto numatytas veiksmų sekas, priklausomai nuo įvykių. Keletas pavyzdžių:

- Pardavimo automatai, atiduodantys prekes, kai įmetama tinkama monetų kombinacija;
- Liftai, pasirenkantys sustojimų eigą priklausomai nuo aukštų, į kuriuos jie buvo iškviesti;
- Šviesoforai, keičiantys savo veikimą priklausomai nuo laukiančių automobilių;
- Užraktai, kuriuose kodo skaičius reikia suvesti tam tikra tvarka.

44. Kompiuterių sparta

Ant stalų sustatyti kompiuteriai. Visi vienu metu gauna uždavinį. Tačiau tik septyni iš jų yra pajėgūs uždavinį išspręsti, bet per skirtingą laiką, pažymėtą minučių skaičiumi (1, 2, ...).



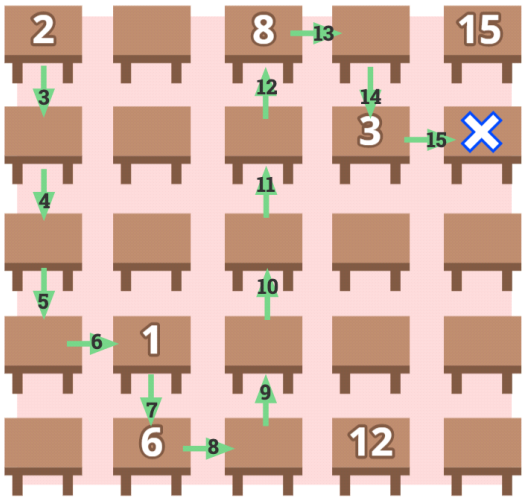
Kompiuterių darbo rezultatai turi pasiekti stalą  ne vėliau kaip po 15 minučių. Yra tik vienas atmintukas rezultatams įrašyti. Jis gali būti perduodamas nuo vieno stalo į gretimą stalą horizontaliai arba vertikaliai (bet ne įstrižai). Perdavimas tarp gretimų stalų užtrunka vieną minutę. Atmintuko kelio pradžia – bet kuris kompiuteris. Kai atmintuką gauna kompiuteris, kuris uždavinį jau išsprendęs, į jį įrašo rezultatą ir siunčia toliau.

Kiek daugiausiai darbo rezultatų gali pasiekti stalą  per ne daugiau kaip 15 minučių?

Paaiškinimas

Teisingas atsakymas: 5

Išbandyti visus galimus maršrutus tarp 7 stalų nelengvas darbas. Gal uždavinį galima supaprastinti radus ir atmetus kai kuriuos neperspektyvius stalus? Tokie bus su kompiuteriais, iš kurių trumpiausias kelias iki stalo X viršys 15 minučių. Taip patikrinus visus 7 stalus atmetame pažymėtus 12 ir 15.



Lieka 5 stalai. Vadinasi, galimų atsakymų intervalą sumažinome iki [0, 5]. Reikia rasti didžiausią. Bandymų keliu radę maršrutą per 5 stalus jau turime atsakymą. Likę nepatikrinti maršrutai atsakymo nekeičia ir jų tikrinti nebereikia.

Tai informatika!

Tai optimizavimo uždavinys. Jame rezultatų, įrašytų į atmintuką, skaičius randamas konstruojant kelią grafu. Kelias yra viršūnių seka grafe. Šiuo atveju stalai atitinka grafo viršūnes, o jų sujungimai – briaunas. Einant keliu nuo pradinio stalo iki galinio, pažymėto X, skaičiuojamas laikas. Jeigu jis neviršija 15 minučių, tai kelias laikomas tinkamu. Iš visų rastų tinkamų kelių išrenkamas tas, kuriame yra daugiausia numeriais pažymėtų jame esančių stalų, t. y., į atmintuką įrašytų rezultatų skaičius. Tas skaičius ir yra uždavinio sprendimas.

Uždavinys gali būti sprendžiamas taikant įvairius metodus. Šiuo atveju pasinaudojama tuo, kad pirmiausia randama, kad du numeruoti stalai negali būti jokiame tinkamame kelyje. Taigi, tinkamų kelių su 6 arba 7 numeruotais stalais negali būti. Todėl pakanka rasti vieną kelią per 5 numeruotus stalus ir laikyti, kad jau rastas optimalus sprendimas.

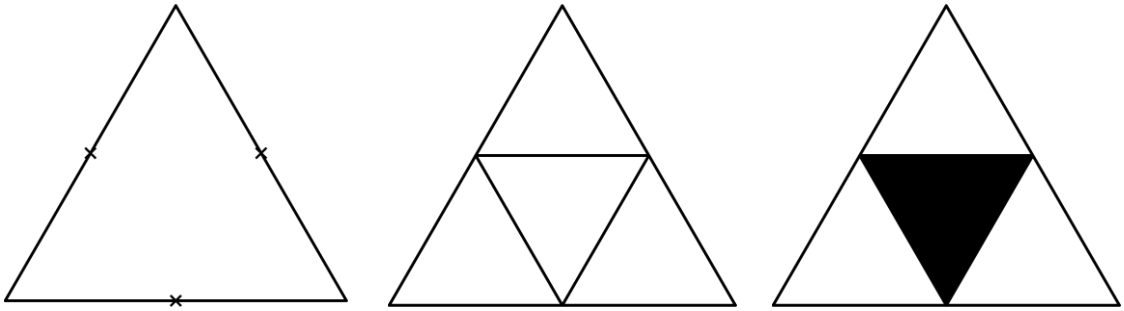
Žmonės, susidūrę su panašiais uždaviniais, pirmiausia bando juos spręsti konstruktyviai. Jie bando sprendimų paieškos erdvę skaidyti į dalis pagal optimumo tikimybę ir atmesti tas dalis, kuriose pavyksta įrodyti, kad ta tikimybė lygi nuliui. Šį metodą aprašė Ailsa Land ir Alison Doig 1960 m. Metodas buvo pavadintas: angliškai *branch and bound*, vokiškai *Verzweigung und Schranke*, prancūziškai *séparation et évaluation*, rusiškai *ветвей и границ*. Lietuviškai gal tiktų skaidymo ir ribojimo. Šis metodas naudojamas sudėtingiems informatikos mokslo uždaviniams spręsti.

45. Sierpinskio trikampis

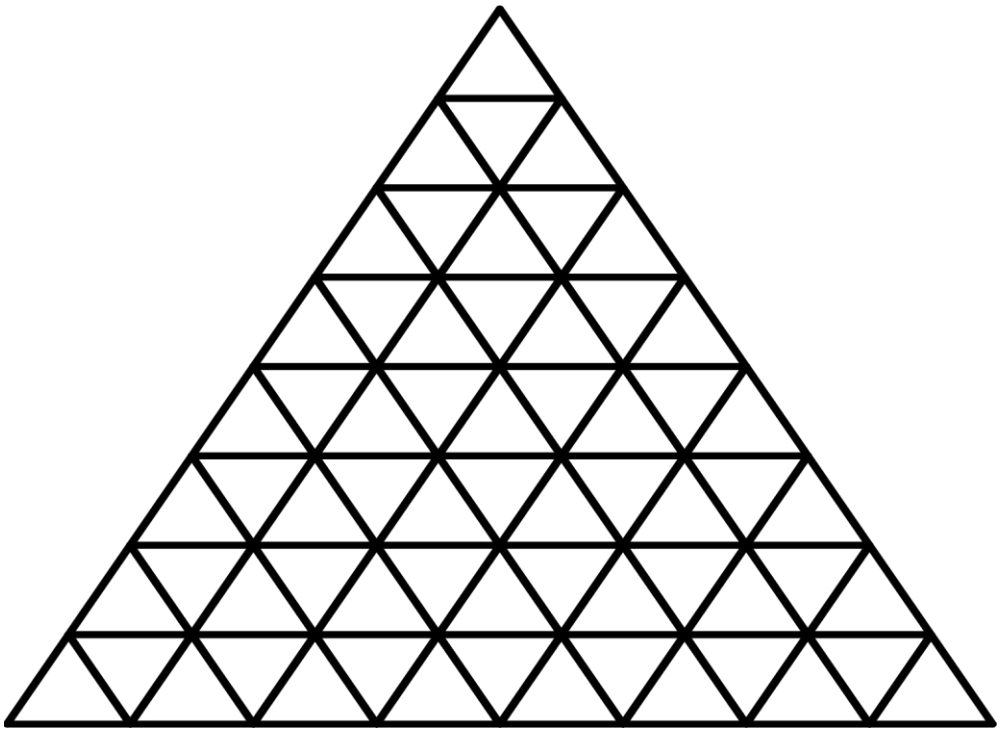
Sierpinskio trikampis pradedamas formuoti nubrėžus lygiakraštį trikampį. Kiekvienam juodai nenuspalvintam trikampiui iteraciškai kartojama žingsnių seka:

1. Žymimi kiekvienos trikampio kraštinės centrai.
2. Šie trys centrai sujungiami formuojant keturis naujus trikampius.
3. Naujai suformuotas centrinis trikampis nuspalvinamas juoda spalva.

Paveiksle parodyti pirmos iteracijos žingsniai:



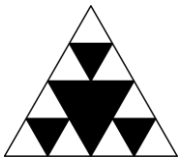
Žingsnių seka pakartota tris kartus. Nuspalvinkite tuos trikampius, kurie turi būti juodi.



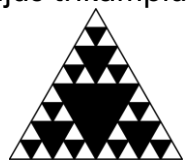
Paaiškinimas

Atlikę pirmą žingsnių sekos iteraciją, gauname vieną juodą trikampį centre:

Pakartojus žingsnius antrą kartą, kiekvienas iš keturių trikampių dalinamas į keturis mažesnius trikampius. Kiekvieno iš tų keturių trikampių centrinis trikampis spalvinamas juoda spalva. Spalvinimas neturi įtakos pradiniam centriniam trikampiui, kadangi jis jau nudažytas juodai. Tačiau atsiranda trys nauji juodi trikampiai:



Atlikę žingsnių seką trečią (paskutinį) kartą, gauname tuos pačius rezultatus kiekvienam iš 16 mažesnių trikampių (arba 9 mažesniems baltiems trikampiams – kiti jau nudažyti juoda spalva). Taip suformuojame 9×4=36 naujus trikampius, 9 iš kurių spalvinami juoda spalva:



Tai vienintelis teisingas sprendimas.

Tai informatika!

Sierpinskio trikampis yra fraktalas, pirmą kartą apibūdintas 1915 m. lenkų matematiko Wacławo Franciszeko Sierpińskio (1882–1969). Panašūs geometriniai objektai (pvz., Kocho snaigė, Mandelbroto aibė) kurie buvo aktyviai tyrinėjami XX amžiuje, ypač atsiradus kompiuteriams, leidusiems automatiškai generuoti ir kartoti daugybę iteracijų. Kadangi dalinimo ir spalvinimo procesas numatytas kartoti be galo daug kartų, jis parodo kai kurias įdomias geometrines savybes, viena kurių yra panašumas į save.

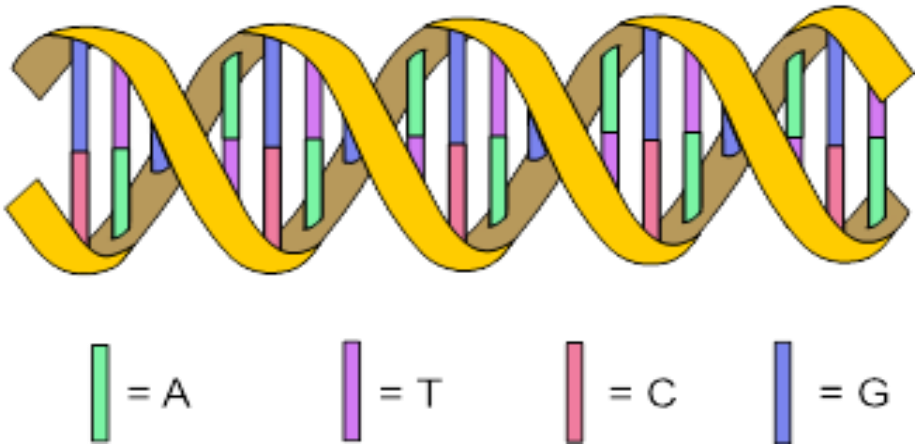
Sierpinskio trikampio konstravimas yra rekursyvus (iš lot. *recurrere*: bėgti atgal, grįžti). Tai reiškia kad tam tikras operacijų rinkinys (tai, ką mes šiame uždavinyje vadinome žingsniais) kartojamas be galo daug kartų, kol neįvyksta tam tikra baigimo sąlyga (pvz., kaip dažnai turi būti kartojamas iteracinis procesas). Tai leidžia kompaktiškai apibrėžti labai sudėtingus objektus.

Rekursija naudojama ir teorinėje, ir praktinėje informatikoje. Sierpinskio trikampis yra tiesiogiai susijęs su tokiais klasikiniais informatikos konceptais, kaip ląstelinis automatas (išpopuliarėjęs Džono H. Konvėjaus (1937–2020 m.) dėka, žaidimas „Gyvenimas“) ar Hanojaus bokštai.

Jei Sierpinskio trikampio formavimas būtų tęsiamas be galo, tam tikru momentu visas trikampis taptų visiškai juodas. Taip yra dėl to, kad atlikus kiekvieną iteraciją tik 75 proc. pradinio balto ploto išlieka baltas. Galima suformuluoti seką, kiek pradinio balto ploto lieka baltos spalvos atlikus tolesnę iteraciją. Seka būtų tokia: a_n = 100 %, 75 %, 75 % × 75 % = 56,25 %, ... Tai geometrinė progresija, kurios $a1 = 1$ ir $q = 0{,}75$. Kadangi geometrinės progresijos, kurių bendras daugiklis $-1 < q < 1$, turi ribą, lygią 0, mūsų atveju riba taip pat yra nulis: $\lim_{n\to\infty}(1 \times 0{,}75n) = 0$.

46. DNR seka

Kiekviena būtybė turi DNR seką, kuri lemia jos genus. DNR seka susideda iš 4 tipų nukleobazių: adenino (A), guanino (G), citozino (C) ir timino (T). DNR gali mutuoti formuodama naujas sekas, kurios skiriasi nuo originalios sekos.



Vormi yra būtybė, kuri gali mutuoti 3 būdais:

1. Pakeitimas: DNR sekoje vienas bazinis komponentas pakeičiamas kitu baziniu komponentu. Pavyzdžiui, AGGTC tampa AGGTA (C keičiamas į A).
2. Pašalinimas: vieno iš bazinių DNR sekos komponentų pašalinimas. Pavyzdžiui, AGGTC tampa AGTC (pašalintas vienas G).
3. Dubliavimas: bazinių komponentų pridėjimas prie DNR sekos. Pavyzdžiui, AGGTC tampa AGGTTC (T dublikatas).

Kurios iš keturių pateiktų Vormi būtybės DNR sekų negalima generuoti vykstant tik 3 genų mutacijoms, jei pradinė Vormi DNR seka yra „GTATCG“?

A. GCAATG
B. ATTATCCG
C. GAATGC
D. GGTAAAC

Paaiškinimas

Teisingas atsakymas: GGTAAAC (D).

Atvejis A: pakeitimas (T į C – GCATCG), pakeitimas (T į A – GCAACG), pakeitimas (C į T – GCAATG)

Atvejis B: pakeitimas (G į A – ATATCG), dubliavimas (T – ATTATCG), dubliavimas (C – ATTATCCG)

Atvejis C: pakeitimas (T į A – GAATCG), pakeitimas (C į G – GAATGG), pakeitimas (G į C – GAATGC)

A, B ir C atvejai yra gaunami vykdant po 3 genų mutacijos žingsnius, o C atvejis reikalauja 4 žingsnių: dubliavimas (G, GGTATCG), dubliavimas (A, GGTAATCG), pakeitimas (T į A, GGTAAACG), pašalinimas (G, GGTAAAC).

Tai informatika!

Ši užduotis reikalauja rasti atstumą tarp vieno žodžio (DNR seka) ir kito.

Reikia išnagrinėti visus būdus ir apskaičiuoti minimalų kiekvienos duotos DNR sekos atstumą. Mažiausias atstumas tarp dviejų žodžių yra vadinamas Levenšteino atstumu.

Levenšteino atstumas taip pat gali būti vadinamas redagavimo atstumu, nes matuojamas mažiausias pavienių ženklų keitimo skaičius. Žinomi redagavimo atstumo operatoriai yra įterpimas, pašalinimas, pakeitimas. Šioje užduotyje įterpimo operacija pakeista dubliavimo operacija.

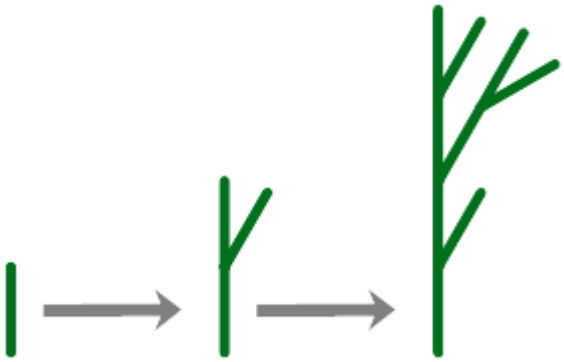
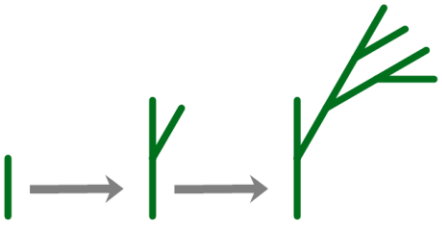
Tarkime, kad DNR seka yra viena eilutė, o mutavusios DNR sekos – kita eilutė. Tada galima išmatuoti Levenšteino atstumą tarp šių dviejų eilučių. Atsakymas yra mutavusi seka, kurios atstumas tarp Levenšteino atstumo nėra lygus 3.

47. Skaitmeniniai medžiai

Skaitmeniniai medžiai auginami pažingsniui pagal tokias taisykles.

Medis pradedamas auginti iš pagaliuko:  . Auginimo taisyklė nusako, kaip pagaliukas keičiamas pagaliukų struktūra. Atliekant kiekvieną žingsnį auginimo taisyklė tuo pat metu taikoma kiekvienam medžio pagaliukui.

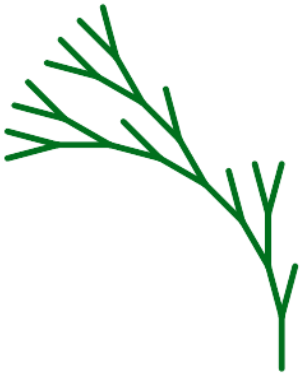
Štai du pavyzdžiai:

Taisyklė	*Pirmieji du žingsniai*	*Taisyklė*	*Pirmieji du žingsniai*
			

Mažos rodyklėlės rodo, kur ir kokia kryptimi pridedamos naujos struktūros.

Pateiktas toks skaitmeninis medis, išaugintas 4-iais žingsniais:

Kuri taisyklė tinka pateiktam medžiui išauginti?



Kuri taisyklė tinka pateiktam medžiui išauginti?

A 

B 

C 

D 

Paaiškinimas

B yra teisingas atsakymas.

A taisyklė: medis auginamas į viršų nuo rodyklėlės, tiesiai aukštyn. Taigi bus tiesus medis su šakomis, nusvirusiomis į kairę.

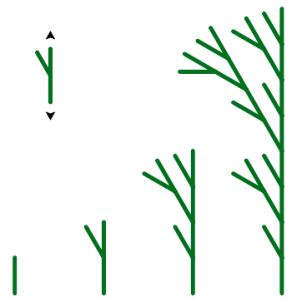
B taisyklė: medis auginamas ant kairiosios atšakos. Taigi visas medis bus palinkęs į kairę.

C taisyklė: medis auginamas ant vidurinės atšakos ir nukreiptas tiesiai aukštyn. Kadangi yra atšakos kairėje ir dešinėje, tai medis bus taisyklinga, simetrinė struktūra.

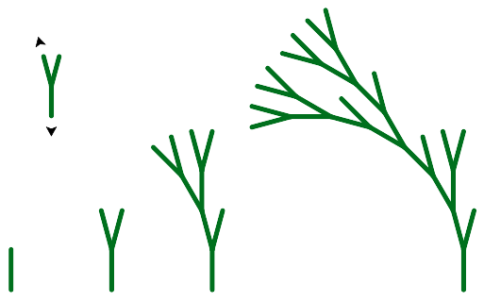
D taisyklė: medis auginamas ant dešiniosios atšakos. Taigi visas medis bus palinkęs į dešinę.

Paveiksliukuose matome, kaip skaitmeniniai medžiai auginami pagal skirtingas taisykles.

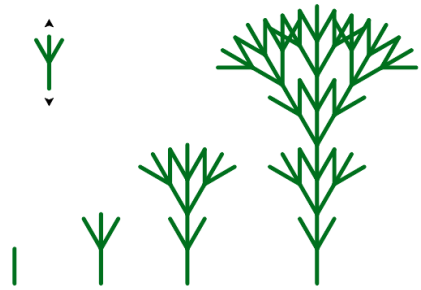
A



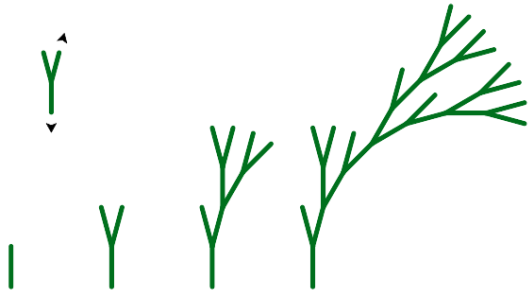
B



C



D



Tai informatika!

Fraktalai dažnai naudojami kompiuterinėje grafikoje gamtovaizdžiams modeliuoti arba įspūdingiems efektams formuoti. Jie taip pat vartojami biologijoje modeliuoti augalų augimą, bakterijų dauginimąsi ir pan.

Lindenmajerio sistemas, vadinamąsias L-sistemas, sukūrė Aristidas Lindenmajeris. Tai galingas formalus metodas augalų ląstelių elgsenai ir augalo vystymosi procesams modeliuoti. L-sistemos gali būti naudojamos generuoti į save panašius fraktalus.

Uždavinyje pateiktas labai paprastas L-sistemos pavyzdys. Jame taikoma tik viena taisyklė ir auginimas pradedamas tik vienu simboliu, o realios sistemos yra žymiai sudėtingesnės.

Medžio auginimo procesas yra pasikartojantis veiksmas, jis paremtas iteracija. Iteracijos dėka programos gali veikti su neribotais kiekiais duomenų. Procesas, kai kiekvienas komponentas keičiamas visa nauja struktūra – medžiu, gali būti geriausiai aprašomas rekursija. Taigi šių skaitmeninių medžių auginimas yra puikus rekursijos pavyzdys.

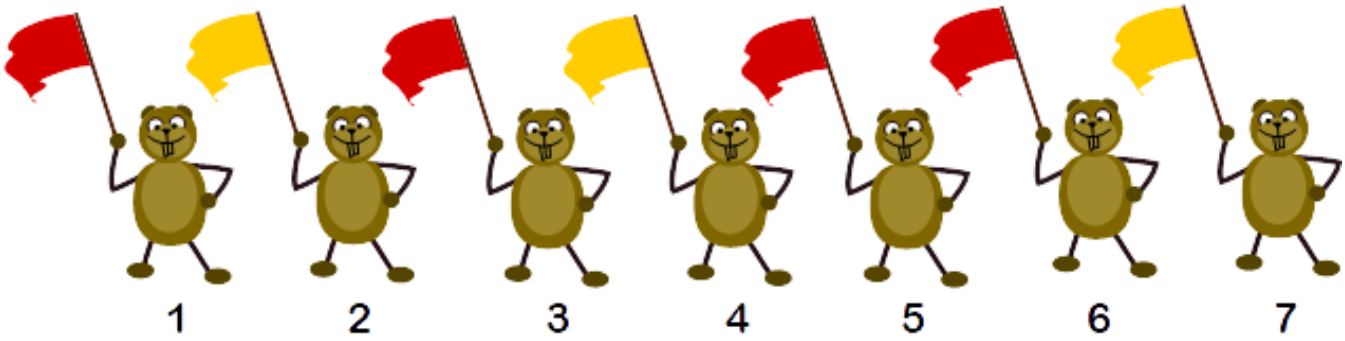
48. Hemingo lemingai

Lemingų karalius Hemingas nori išsiųsti pranešimą savo karalienei, gyvenančiai kitoje pilyje. Tam jis pasirenka keturis lemingus pranešimui perduoti ir kiekvienam iš jų įteikia po vėliavą – raudoną ar geltoną, – spalva priklauso nuo siunčiamo pranešimo turinio. Tačiau karalius nerimauja, kad kelionėje neatsitiktų kas netikėto, todėl pasirenka dar tris lemingus-pagalbininkus ir įteikia jiems po vėliavą remdamasis tokia taisykle:

Kiekvienas lemingas-pagalbininkas gauna raudoną vėliavą, jei lemingų, kuriems jis padeda, bendras turimų raudonų vėliavų skaičius nelyginis, o jei lyginis arba tokių vėliavų nėra, tai gauna geltoną vėliavą.

- 5-as lemingas padeda 1, 2 ir 3 lemingams.
- 6-as lemingas padeda 1, 2 ir 4 lemingams.
- 7-as lemingas padeda 2, 3 ir 4 lemingams.

Kai lemingai atvyko į pilį, karalienė pamatė tokį vaizdą:



Vienas lemingas pametė savo vėliavą. Norėdamas pasitaisyti, jis greitai pagamino naują vėliavą. Tačiau jo nauja vėliava neteisingos spalvos. Kuris lemingas pametė savo vėliavą?

A. Nauja vėliava yra teisingos spalvos.
B. 1-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.
C. 2-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.
D. 3-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.
E. 4-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.
F. 5-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.
G. 6-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.
H. 7-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.

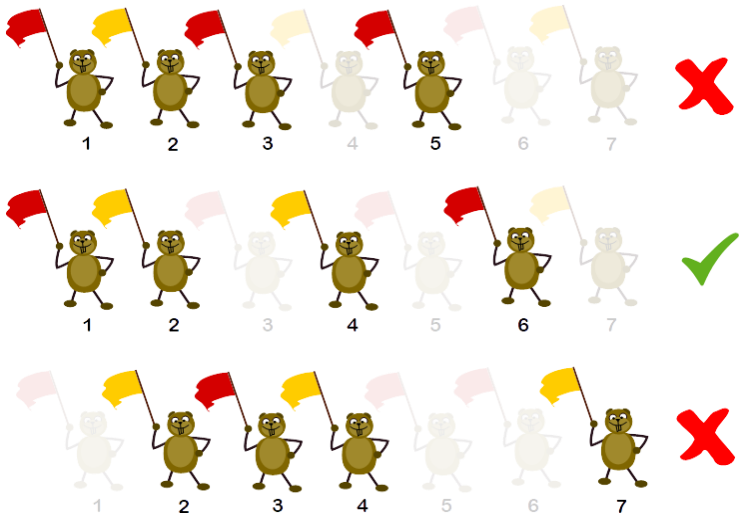
Paaiškinimas

3-as lemingas pametė savo vėliavą, o jo nauja vėliava neteisingos spalvos.

Nagrinėkime padedančius lemingus ir lemingų, kuriems jie padeda, grupes. Pažymėkime grupę, kuriai lemingas padeda ir jį patį lemingo-pagalbininko numeriu. Pavyzdžiui, 5-ą grupę sudaro 5-as lemingas ir lemingai, kuriems jis padeda, t. y., 1, 2 ir 3 lemingai.

Iš taisyklių, pagal kurias karalius įteikė vėliavas lemingas-pagalbininkams, galime pastebėti, kad prieš prasidedant kelionei raudonų vėliavų bendras skaičius, turimas bet kurioje grupėje, turi būti lyginis. Iš tikrųjų, jei lemingai 1, 2 ir 3 neša nelyginį raudonų vėliavų skaičių, tai 5-as lemingas-pagalbinnkas gauna raudoną vėliavą, o tai reiškia, kad bendras šioje grupėje esančių raudonų vėliavų skaičius yra lyginis. Jei 1, 2 ir 3 lemingų raudonų vėliavų skaičius būtų lyginis, 5-as lemingas gautų geltoną vėliavą, o grupelės raudonų vėliavų skaičius taip pat būtų lyginis. Tas pats galioja 6 ir 7 grupėms.

Dabar patikrinkime, ar ši taisyklė galioja visoms grupėms, joms atvykus į karalienės pilį.



Matome, kad taisyklė galioja ne visoms grupėms. Tai reiškia, kad lemingas, pametęs vėliavą, supainiojo spalvą ir pagamino naują vėliavą neteisingos spalvos.

Jei neteisinga vėliava būtų laikoma pagalbininko, tik viena grupė neatitiktų minėtos savybės – ta grupė, kurioje yra pagalbininkas. Kadangi savybė negalioja daugiau kaip vienai grupei, tai galime daryti išvadą, kad vėliavą pametė vienas iš šių lemingų: 1, 2, 3 arba 4. Jau žinome, kad vėliavą pametęs lemingas pakeitė vėliavą nauja neteisingos spalvos. Kiekviena grupė, į kurią įeina šis lemingas, neatitiktų taisyklės. Reikia rasti lemingą, kuris yra grupėse, kurioms negalioja taisyklė (5 ir 6 grupė), bet nėra grupėje, kuriai taisyklė galioja (7 grupė). 3-as lemingas vienintelis tenkina šiuos reikalavimus.

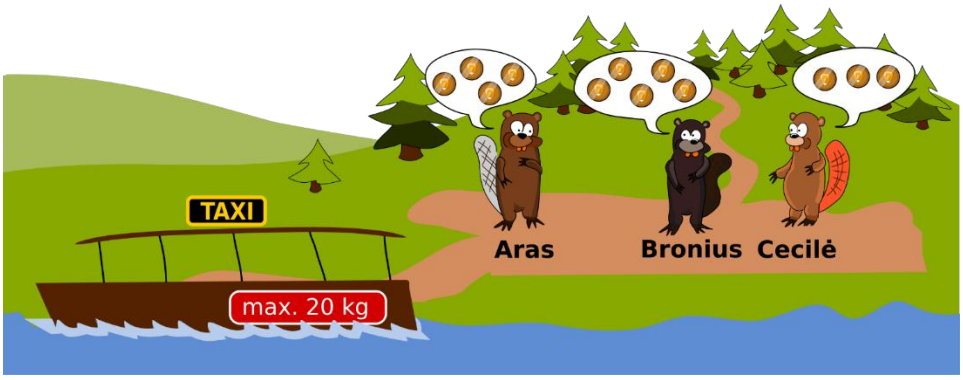
Tai informatika!

Kiekvienas klaidų taisymo kodas turi trūkumų. Kai kurie kodai skirti tik nustatyti, ar yra pakeitimų pranešimo bituose neatstatant pradinių reikšmių. Kai kurie kodai veikia tik jei ne didesnis kaip tam tikras skaičius bitų buvo pakeistas. Pavyzdžiui, šio uždavinio kodas veikia tik jei vienas iš lemingų sukeitė savo vėliavos spalvą. Jei tai būtų padarę du lemingai, karalienė negalėtų rasti ir ištaisyti klaidų pranešime. Jei to nebūtume žinoję, gali būti atvejų, kad pranešimas atkeliavo teisingas, net jei lyginumai yra neteisingi (pvz., jei ir 5, ir 7 lemingai pakeistų savo vėliavų spalvas). Darant prielaidą, kad mažai tikėtina, jog daug bitų būtų pakeista siuntimo metu (paprastai tik vienas ar keli bitai), klaidų taisymo kodai, tokie kaip Hemingo kodas, pakankamai gerai veikia. Tačiau realiose komunikacijų sistemose situacijos sudėtingesnės, negu šiame uždavinyje, todėl kuriami sudėtingesni klaidų taisymo algoritmai.

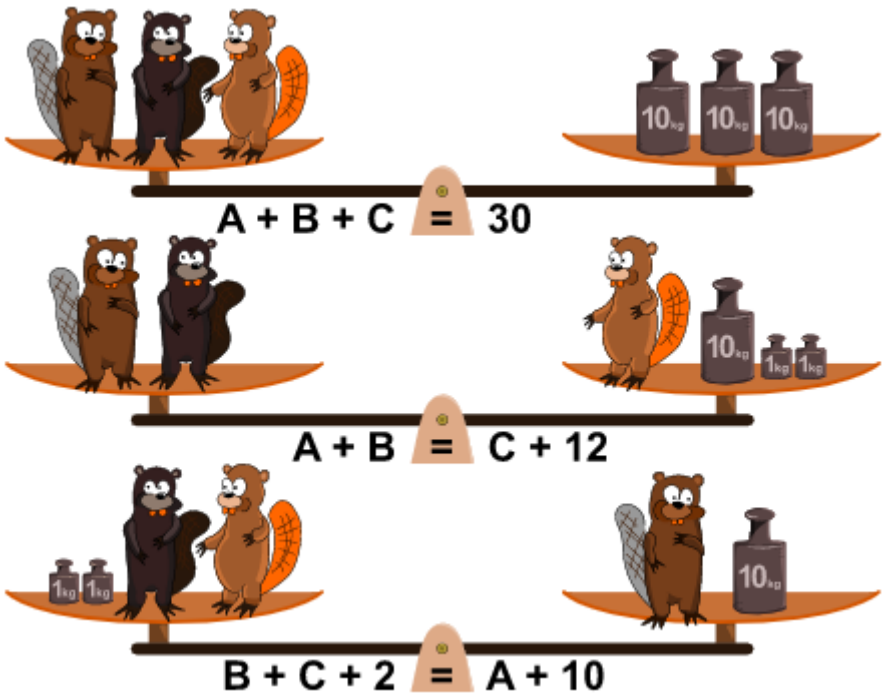
Panašūs algoritmai taikomi ir kitose srityse, pavyzdžiui, kuriant brūkšninius kodus, kuriuos matome ant įvairių prekių, identifikavimo numerius, pasų numerius.

49. Vandens taksi

Trys bebrai Aras, Bronius ir Cecilė nori nuvykti į miestą su vandens taksi.



Yra tik vienas vandens taksi. Aras už kelionę moka 4 monetas (4x). Bronius sutinka už kelionę mokėti 5 monetas (5x), o Cecilė - 3 monetas (3x). Taksi vairuotojas nori gauti kuo didesnį užmokestį (pelną), tačiau jis valtimi gali saugiai plukdyti grupę keleivių, kurių bendras svoris yra ne didesnis kaip 20 kg. Todėl prieš kelionę keleiviai turi pasisverti. Svėrimų rezultatai parodyti paveikslėlyje ir pagal tuos rezultatus priimamas sprendimas.



Koks yra pats pelningiausias taksisto sprendimas, neperkraunant jo valties?

A) Tik Bronius

B) Aras ir Bronius

C) Bronius ir Cecilė

D) Aras ir Cecilė

E) visi trys: Aras, Bronius ir Cecilė

Paaiškinimas

Pirmiausia reikia sužinoti atskirų bebrų svorį, kad būtų galima išvardyti visus galimus sprendimus (visus poaibius, kurių bendras svoris ne didesnis kaip 20 kg). Vienas iš sprendimo būdų yra atspėti bebrų svorį. Visi trys sveria 30 kg, o tai vidutiniškai yra 10 kg bebrui. Antrasis matavimas A + B = C + 12 kg suteikia užuominą, kad C sveria tikriausiai mažiau nei 10 kg. Trečias matavimas rodo, kad A sveria daugiau nei 10 kg. Galima bandyti paimti C = 9 kg ir A = 11 kg. Įstačius reikšmes nustatoma, kad tai yra galimo sprendimo dalis, ir apskaičiuojama, kad B = 10 kg.

Griežtas būdas yra išspręsti tiesinių lygčių sistemą, gautą atlikus tris svėrimus:

I. A + B + C = 30 kg

II. A + B = C + 12 kg

III. B + C + 2 kg = A + 10 kg

Tai galima pertvarkyti, kad A, B ir C būtų vienoje pusėje:

I. A + B + C = 30 kg

II. A + B - C = 12 kg

III. - A + B + C = 8 kg

Apibendrinus I ir II mes gauname:

IV. 2A + 2B = 42 kg / 2

A + B = 21 kg

Apibendrinus II ir III mes gauname:

V. 2B = 20 kg / 2

B = 10 kg

Žinodami, kad A + B = 21 kg ir B = 10 kg, gauname A = 11 kg.

Įterpdami A ir B į bet kurią iš trijų lygčių, gauname C = 9 kg.

Dėl visų keleivių bendro svorio apribojimo taksi vairuotojas negali pasiimti visų trijų bebrų. Be to, jis negali paimti ir A, ir B, nes A + B = 21 kg. Todėl lieka 5 variantai, tenkinantys 20 kg apribojimą: paimti A ir C arba paimti B ir C arba po vieną bet kurį iš bebrų.

Galiausiai mes patikriname, kuris variantas yra pelningiausias. Paimti bet kurį bebrą yra mažiau pelninga nei paimti du kartu. A ir C kartu sumokės 7 monetas, B ir C kartu sumokės 8 monetas. Todėl teisingas atsakymas yra C variantas – su taksi keliauja Bronius ir Cecilė.

Tai informatika!

Diskretinė optimizacija yra svarbi algoritmų teorijos šaka. Atsižvelgus į galimų sprendimų apribojimus ir išbandžius visų galimų sprendimų kainas, reikia pasirinkti sprendimą su didžiausia arba minimalia kaina. Čia pateikta užduotis yra garsiojo kuprinės uždavinio apibendrinimo pavyzdys. Viena iš pagrindinių kompiuterių mokslininkų kompetencijų yra mokėti išvardyti visus galimus tam tikro uždavinio sprendimus, kad juos įvertintumėte ir palygintumėte su kai kuriais kriterijais. Šia kompetencija grindžiama visa sprendimo logika.

Atkreipkite dėmesį, kad kiekvieną „NP-sunkumo“ optimizavimo uždavinį galima išreikšti kaip tiesinį programavimo uždavinį, t. y., kaip uždavinį su apribojimais, kuriuos suteikia sveikųjų skaičių tiesinių lygčių sistema ir tiesinė tikslinė funkcija. Mūsų vandens taksi yra būtent tokio pobūdžio uždavinys. „Vandens taksi“ yra ypatingas tiesinio programavimo atvejis su papildoma užduotimi išsiaiškinti bebrų svorį.

50. Pinigų maišeliai

Bebras Matas deda savo metalines monetas į vandeniui nepralaidžius maišelius. Šiandien jis turi 63 vieno bebro monetas ir nori jas sudėti į maišelius taip, kad galėtų mokėti bet kurią sumą nuo 1 bebro iki 63 bebrų tiesiog paduodamas vieną ar daugiau maišelių ir jų neatidarydamas.

Koks reikalingas mažiausias maišelių skaičius?

Paaiškinimas

Atsakymas: 6. Matas turi sudėti į 6 maišelius tokį 1 bebro monetų skaičių: 1, 2, 4, 8, 16, 32.

Akivaizdu, kad 1+2+4+8+16+32=63, ir jis gali mokėti bet kurią sumą nuo 1 iki 63 bebrų neatidarydamas maišelių. Pavyzdžiui, norėdamas sumokėti 13 bebrų, jis gali įteikti tris maišelius su 1, 4 ir 8 bebro monetomis.

Lentelėje parodyta, kokį bebro monetų skaičių nuo 1 to 63 galima sumokėti naudojant kai kuriuos iš šių 6 maišelių (jų neatidarant).

Kiekviename maišelyje esantis monetų skaičius pateiktas lentelės antraštinėje eilutėje. Sankirtos langelyje parašytas skaičius 1, jei Matas mokėdamas sumą naudoja atitinkamą maišelį. Jei nenaudoja, langelyje yra 0.

Bebro monetų skaičius	*Maišelis*						*Bebro monetų skaičius*	*Maišelis*					
	32 B	*16 B*	*8 B*	*4 B*	*2 B*	*1 B*		*32 B*	*16 B*	*8 B*	*4 B*	*2 B*	*1 B*
1	0	0	0	0	0	1	33	1	0	0	0	0	1
2	0	0	0	0	1	0	34	1	0	0	0	1	0
3	0	0	0	0	1	1	35	1	0	0	0	1	1
4	0	0	0	1	0	0	36	1	0	0	1	0	0
5	0	0	0	1	0	1	37	1	0	0	1	0	1
6	0	0	0	1	1	0	38	1	0	0	1	1	0
7	0	0	0	1	1	1	39	1	0	0	1	1	1
8	0	0	1	0	0	0	40	1	0	1	0	0	0
9	0	0	1	0	0	1	41	1	0	1	0	0	1
10	0	0	1	0	1	0	42	1	0	1	0	1	0
11	0	0	1	0	1	1	43	1	0	1	0	1	1
12	0	0	1	1	0	0	44	1	0	1	1	0	0
13	0	0	1	1	0	1	45	1	0	1	1	0	1
14	0	0	1	1	1	0	46	1	0	1	1	1	0
15	0	0	1	1	1	1	47	1	0	1	1	1	1
16	0	1	0	0	0	0	48	1	1	0	0	0	0
17	0	1	0	0	0	1	49	1	1	0	0	0	1
18	0	1	0	0	1	0	50	1	1	0	0	1	0
19	0	1	0	0	1	1	51	1	1	0	0	1	1
20	0	1	0	1	0	0	52	1	1	0	1	0	0
21	0	1	0	1	0	1	53	1	1	0	1	0	1
22	0	1	0	1	1	0	54	1	1	0	1	1	0
23	0	1	0	1	1	1	55	1	1	0	1	1	1
24	0	1	1	0	0	0	56	1	1	1	0	0	0
25	0	1	1	0	0	1	57	1	1	1	0	0	1
26	0	1	1	0	1	0	58	1	1	1	0	1	0
27	0	1	1	0	1	1	59	1	1	1	0	1	1
28	0	1	1	1	0	0	60	1	1	1	1	0	0
29	0	1	1	1	0	1	61	1	1	1	1	0	1
30	0	1	1	1	1	0	62	1	1	1	1	1	0
31	0	1	1	1	1	1	63	1	1	1	1	1	1
32	1	0	0	0	0	0							

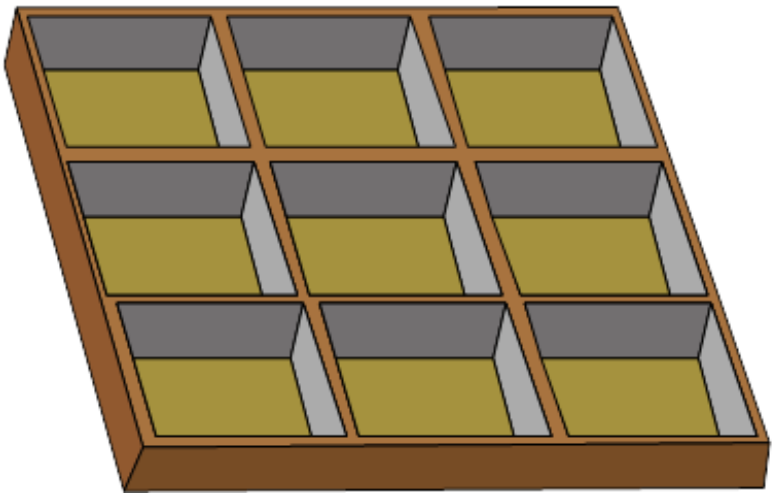
Tai informatika!

Šis uždavinys – apie dvejetainę skaičiavimo sistemą. Matematikoje ir informatikoje dvejetainiai skaičiai nagrinėjami skirtingai. Matematikoje labiausiai nagrinėjamos tokių skaičių savybės, o informatikoje – ieškoma jų taikymų. Kompiuteriuose naudojama dvejetainė sistema įvairios rūšies informacijai pateikti: dokumentams, paveikslams, garso ir vaizdo įrašams, skaičiams ir net programoms, kurias naudojame. Visa tai užkoduojama dvejetainiais skaičiais – bitais (angl. *binary digits - bits*). Vienas bitas negali koduoti daug informacijos, todėl bitai grupuojami siekiant pateikti didesnį informacijos kiekį. Kompiuteriuose bitai paprastai jungiami į klasterius po aštuonis bitus, vadinamus baitais. Vienas baitas gali koduoti skaičius nuo 0 iki 255.

Šiame uždavinyje Matas naudoja 6 dvejetainio skaičiaus skaitmenis (maišelius) visiems skaičiams nuo 1 iki 63 pateikti.

51. Devyni stiklo kamuoliukai

Henrika turi dėžutę su 9 skyreliais ir 9 stiklo kamuoliukus:



Henrika paima bet kokį skaičių stiklo kamuoliukų (nuo 0 iki 9) ir juos sudeda dėžutėje pagal šias taisykles:

- viename skyrelyje gali būti tik vienas kamuoliukas;
- kiekvienoje eilutėje esančių kamuoliukų skaičius turi būti lyginis;
- kiekviename stulpelyje esančių kamuoliukų skaičius turi būti lyginis.

Keliais skirtingais būdais Henrika gali sudėti stiklo kamuoliukus į dėžutę?

A. 12

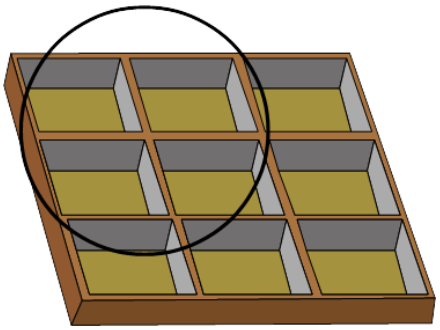
B. 16

C. 64

D. 512

Paaiškinimas

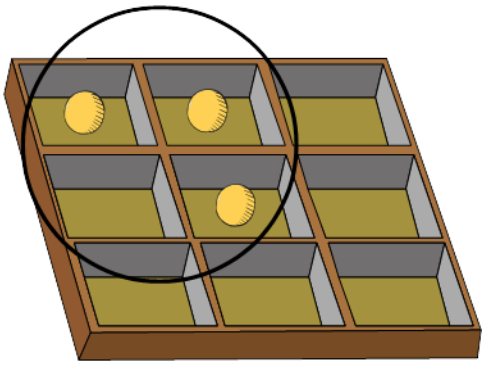
Atsakymas: 16.

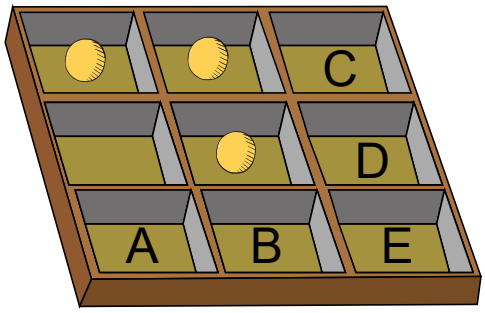


Pirmiausia patyrinėkime tik 2x2 dėžutės dalį. Šioje dalyje yra keturi skyreliai ir kiekviename iš jų arba bus įdėtas kamuoliukas arba ne. Taigi, yra $2^4 = 16$ skirtingų variantų sudėti stiklo kamuoliukus į šią dėžutės dalį.

Svarbu pastebėti, kad po to, kai Henrika nusprendžia, kaip užpildyti šią dėžutės dalį, ji neturi pasirinkimo, kaip užpildyti likusius dėžutės skyrelius. Laikantis sąlygos, kad eilutėse ir stulpeliuose turi būti lyginis skaičius stiklo kamuoliukų, kiekviename iš likusių skyrelių Henrika turės arba įdėti kamuoliuką arba ne.

Pavyzdžiui, tarkime, kad Henrika užpildė 2x2 dalį taip, kaip parodyta paveikslėlyje:





Kadangi pirmame stulpelyje yra tik vienas kamuoliukas, Henrika turi pridėti dar vieną kamuoliuką A skyrelyje, kad stulpelio kamuoliukų skaičius būtų lyginis. Antrajame stulpelyje jau yra lyginis kamuoliukų skaičius, taigi Henrika privalo palikti B skyrelį tuščią. Panašiai samprotaujant Henrika privalo palikti C skyrelį tuščią, o D ir E skyreliuose padėti po kamuoliuką. Taigi, Henrika gali sudėti stiklo kamuoliukus į dėžutę 16 skirtingų būdų.

Tai informatika!

Perduoti duomenis saugiai ir tiksliai yra svarbi informatikos mokslo užduotis. Vienas iš būdų užtikrinti, kad duomenys perdavimo metu nebuvo prarasti arba pakeisti, yra atlikti lyginumo patikrą (angl. *parity check*). Taikant šį metodą lyginis bitas yra pridedamas siunčiamo pranešimo pabaigoje. Kai pranešimą perskaito gavėjas, lyginio bito reikšmė iš siunčiamų duomenų apskaičiuojama iš naujo ir, jei ji neatitinka gauto pranešimo lyginio bito reikšmės, galima daryti išvadą, kad siunčiamas pranešimas buvo iškraipytas.

Šioje užduotyje būtent apatinė eilutė ir dešinysis stulpelis ir atlieka lyginių bitų vaidmenį. Jei dėžutė su stiklo kamuoliukais būtų siunčiama kaip pranešimas, gavėjas galėtų patikrinti eilučių ir stulpelių sumas ir, jei jos nėra lyginės, pateikti siuntėjui pranešimą, kad persiunčiant duomenis įvyko klaida.

52. Vėliavos

Kompiuterių paveikslai sudaryti iš taškelių, vadinamų pikseliais. Kiekvieno pikselio spalva koduojama atskirai.

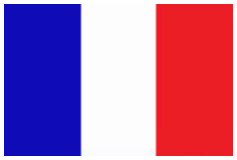
Taškinės grafikos formatas GIW skirtas duomenims glaudinti pagal štai tokį algoritmą:

- koduojamas kiekvienas pikselis;
- kiekviena spalva koduojama trimis raidėmis;
- tos pačios spalvos pikselių seka koduojama skliaustuose nurodant spalvos kodo santrumpą ir tos spalvos pikselių skaičių.

Pavyzdys. Pikselių eilutė, kurios pradžioje yra 20 žalių pikselių, po to – 13 baltų, koduojama šitaip: (žal,20)(bal,13).

Pateikti keturių vėliavų paveikslai. Jie sudaryti iš tiek pat pikselių eilučių ir tiek pat pikselių juose. Paveikslai suglaudinami GIW formatu. Kuo mažesnis skliaustelių skaičius užkoduotame faile, tuo failas lengvesnis.

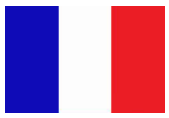





Išrikiuokite vėliavų paveikslus nuo didžiausio iki mažiausio suglaudinto failo.

Paaiškinimas

Atsakymas:

Prancūzija	
Vokietija	
Čekija	
Švedija	

Jei visa eilutė sudaryta iš vienos spalvos, tai joje yra tik vieni skliausteliai, aprašantys visą eilutę. Kiekvienoje eilutėje yra tiek skliaustų, kiek spalvų pasikeitimų. Pavyzdžiui, Vokietijos vėliavoje visos eilutės sudarytos iš vienos spalvos pikselių, todėl suglaudintame faile kiekviena eilutė bus aprašome tik vienais skliaustais.

Čekijos vėliavoje kiekvienoje eilutėje spalvos keičiasi tik vieną kartą, todėl jos suglaudintame faile visose eilutėse bus po dvi skliaustelių poras.

Prancūzijos vėliavos kiekvienoje eilutėje spalva keičiasi po du kartus: mėlyną keičia balta, o baltą – mėlyna. Todėl kiekvienos eilutės kodavimui reikės trijų skliaustelių porų.

Švedijos vėliava yra sudėtingesnė. Jos viduryje spalvos nesikeičia, todėl šią sritį aprašančių eilučių kodavimui užteks vienų skliaustelių. Kitose srityse spalvos keičiasi po du kartus: geltoną keičia mėlyna, o mėlyną – geltona. Kiekvienos šios srities eilutės kodavimui reikės trijų skliaustelių porų. Akivaizdu, kad Švedijos vėliavos kodavimui reikės daugiau skliaustelių nei Vokietijos, bet mažiau nei Prancūzijos.

Palyginkime Švedijos ir Čekijos vėliavas.

Jei Švedijos vėliavoje būtų toks pat skaičius vienos spalvos eilučių kaip ir eilučių su du kartus besikeičiančia spalva, tai tuomet kiekvienai eilutei vidutiniškai tektų po du skliaustelius (tiek pat, kaip ir Čekijos vėliavoje). Kadangi viduryje vėliavos esanti geltona juosta yra siauresnė už likusią dalį, tai eilučių su du kartus besikeičiančia spalva yra daugiau, nei vienos spalvos. Taigi, Švedijos vėliavos suglaudintas faile bus daugiau skliaustelių nei Čekijos vėliavos faile.

Tai informatika!

Duomenų glaudinimas – labai svarbi informatikos dalis, nes leidžia sumažinti užimtą kompiuterio atminties dalį, bendrą tinklo srautą ir duomenų perdavimo laiką. Duomenų glaudinimo algoritmai gali sumažinti kaštus, o tai labai svarbu ieškant efektyvių algoritmų glaudinant nuotraukas, garso ir vaizdo failus, nes jie užima ypatingai daug atminties.

Uždavinyje pateiktas duomenų glaudinimas vadinamas pasikartojimų kodavimu (angl. *run length encoding*, RLE). Apie jo esmę galima daugiau sužinoti vaizdo įraše: https://www.youtube.com/watch?v=ypdNscvym_E

Kuriame Lietuvos ateitį
2014–2020 metų Europos Sąjungos fondų investicijų veiksmų programa

Vilniaus universitetas

Informatinio mąstymo uždavinių rinkiniai sukurti įgyvendinant projektą „Aukštųjų mokyklų tinklo optimizavimas ir studijų kokybės gerinimas Šiaulių universitetą prijungiant prie Vilniaus universiteto“ (Nr. 09.3.1-ESFA-V-738-03-0001), finansuojamą iš Europos socialinio fondo lėšų pagal 2014–2020 metų Europos Sąjungos fondų investicijų veiksmų programos 9 prioriteto „Visuomenės švietimas ir žmogiškųjų išteklių potencialo didinimas“ įgyvendinimo priemonę Nr. 09.3.1-ESFA-V-738 „Aukštųjų mokyklų tinklo tobulinimas“.

„Bebro“ konkurso užduočių rinkinys tinka Mokyklos pedagogikos studijų programos moduliui „Informatikos didaktika“. Studentai, būsimi mokytojai, nagrinėdami „Bebro“ uždavinius susipažįsta su įvairiais informatikos konceptais, gilinasi į sudėtingesnes informatikos sąvokas ir mokosi paaiškinti.