ISSN 2783-7599

b·e·b·r·a·s

Informatikos ir informatinio mąstymo uždavinių rinkys

Nr. 9

ŠVIETIMO, MOKSLO IR SPORTO MINISTERIJA

VILNIAUS UNIVERSITETAS ·1579· UNIVERSITAS VILNENSIS

Bebras
International Challenge on Informatics and Computational Thinking

Šiame rinkinyje pateikiami 2021 metų XVIII 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.

Tarptautinis „Bebro“ iššūkis vykdomas daugiau kaip šešiasdešimtyje šalių. Tai vienas iš būdų mokyti informatikos neformaliai: mokiniai, spręsdami įdomius, informatikos konceptais grįstus uždavinius, susipažįsta su informatikos sąvokomis, ugdosi gebėjimą taikyti jas praktiškai, uždavinių sprendimus aptaria su bendraamžiais ir mokytojais.

„Bebro“ iššūkio tikslas – skatinti informatinį mąstymą ne tik tarp įvairaus amžiaus mokinių, bet ir tarp mokytojų ar visuomenėje. Vyresniųjų klasių mokinimas skirti „Bebro“ uždaviniai tinka ir informatikos mokytojams, kuriems svarbu pristatyti informatikos ar algoritmų konceptus patraukliai, vaizdžiomis užduotimis.

Dėkojame Daumilui Ardickui, dr. Vaidui Giedrimui, Nojui Gudinavičiui, Karoliui Jasučiui, dr. Tatjanai Jevsikovai, dr. Eimučiui Karčiauskui, Audronei Klupšaitei, Linui Kurtinaičiui, Alvidai Lozdienei, Vaidai Masiulionytei-Dagienei, Modestui Rimkui, dr. Broniui Skūpui, dr. Gabrielei Stupurienei, Tomui Šiauliui, dr. Sigitai Turskienei, Kunigundai Zubytei, Rimantui Žakauskui, talkinusiems verčiant ir adaptuojant uždavinius. Taip pat dėkojame tarptautinei „Bebro" konkurso 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, ugdyti 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“ konkurso 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: surinkta daugybė duomenų, atliekami tyrimai 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 juose slypi, parodoma, kaip galima eiti toliau ir giliau.

Mokiniai, spręsdami įdomius, informatikos konceptais grįstus uždavinius, susipažįsta su informatikos sąvokomis, 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 pajvairinti. Mokslininkai kurdami uždavinius stengiasi perteikti esminius informatikos mokslo principus. „Bebro“ bendruomenę labiausiai

vienija patrauklių, įdomių informatikos uždavinių paieška, informatikos konceptų išreiškimas žaidybinėmis užduotimis.

XVII „Bebro“ konkurso seminaras-dirbtuvės vyko 2021 m. gegužės 17–21 d. nuotoliniu būdu, naudojant įvairias technologines priemones: „Zoom“ internetiniams seminarams, „Zulip“ pokalbiams, „BigBlueButton“ vaizdo konferencijoms, SVN užduočių versijoms atnaujinti, taip pat internetines biuro priemones ir kita. Renginyje dalyvavo 69 šalių delegacijos.

XVII tarptautinio „Bebro“ seminaro-dirbtuvių, vykusių nuotoliniu būdu, dalyviai

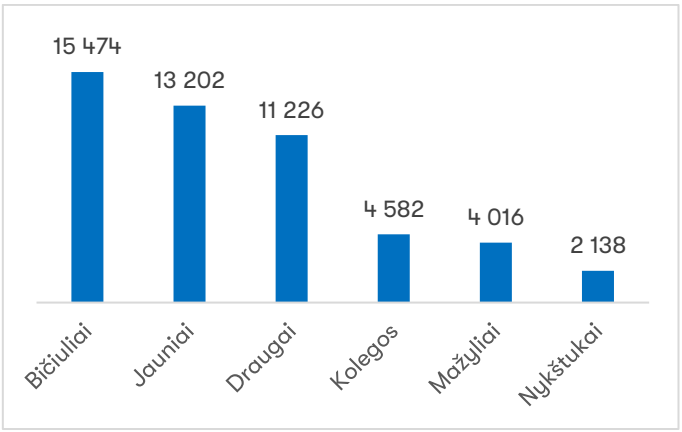
Nuo 2004 metų „Bebro“ bendruomenė sukūrė šimtus užduočių. Dauguma jų yra suformuluotos kaip interaktyvūs ir (arba) atvirieji klausimai. Tačiau net ir tada, kai atsakymą reikia pasirinkti iš sąrašo, nėra vienintelio būdo pateikti sprendimą. Užduotys turi būti įdomios ir patrauklios, atitikti konkurso dalyvių amžių, o sprendimui turėtų prireikti vidutiniškai trijų minučių. Be to, kadangi konkurso tikslas yra paremti mokyklas, kuriose informatikos gali būti mokoma gana skirtingu lygiu, užduotys turėtų būti nepriklausomos nuo specifinių techninių žinių, reikia vengti vartoti žargoną. Tiesą sakant, „Bebro“ užduotyse daugiausia dėmesio skiriama tai informatikos daliai, kurią turėtų išmanyti visi, kas nori suprasti informatikos, kaip mokslo, pagrindus ir tapti ne tik sumaniu technologijų naudotoju, bet ir mokytis

moderniais metodais spręsti įvairiausias realaus gyvenimo problemas, tapti naujų technologijų kūrėju.

„Bebro“ bendruomenė užduotis papildo komentarais apie kiekvienoje užduotyje slypinčią kertinę informatikos sąvoką (skyrelis „Tai informatika!“). Kai kurios „Bebro“ konkursą vykdančios šalys pateikia papildomos metodinės medžiagos mokytojams (paprastai tai daro išleisdamos užduočių bukletus ar publikuodamos jas interneto svetainėse), tačiau daugeliu atvejų mokytojai tiesiog po konkurso aptaria užduotis su mokiniais. Kita vertus, „Bebro“ užduočių visuma gali tapti reikšmingu ištekliumi mokyti informatikos ir informatinio mąstymo, kadangi užduotys yra lengvai pasiekiamos ir jų turinys aiškiai identifikuotas. Jos gali būti naudojamos įvairiai edukacinei veiklai, naujiems konkursams organizuoti.

2021 metų „Bebro“ uždavinius sprendė 3 033 518 mokinių 57-iose valstybėse. Lietuvoje tais metais konkurse dalyvavo 50 638 mokiniai:

- 2 138 nykštukai (1–2 klasės),
- 4 016 mažyliai (3–4 klasės),
- 15 474 bičiuliai (5–6 klasės),
- 11 226 draugai (7–8 klasės),
- 13 202 jauniai (9–10 klasės),
- 4 582 kolegos (11–12 klasės).



Konkurso dalyvių skaičius Lietuvoje 2021 m.

2021 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 už uždavinį 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 galite pasiekti visų dalyvaujančių šalių svetaines – spustelėkite šalies vėliavėlę).
- Lietuvos „Bebro“ svetainė: bebras.lt
- Konkurso sistema – vadinamasis „Bebro“ varžybų laukas: lt.bebras.lt

Lentelėje pateikiamas šio rinkinio uždavinių skirstymas pagal amžiaus grupes.

Nr.	*Uždavinio pavadinimas*	*Uždavinio identifikatorius*	*Nykštukai*	*Mažyliai*	*Bičiuliai*	*Draugai*	*Jauniai*	*Kolegos*
1	Mykolo antspaudai	2021-HU-05c	6	6				
2	Tilto statyba	2021-PK-07	6	6				
3	Monetų dėlionė	2021-CZ-06	6	6				
4	Rask gyvūno vardą	2021-LT-04b	6					
5	Futbolo marškinėliai	2021-IE-04	9	6				
6	Raidės	2021-HR-03	9		6			
7	Kaliausė	2016-LT-01	9					
8	Roboto kelias	2016-LT-04	9					
9	Vėrinys	2021-SK-01	12	9	6			
10	Žvilgsnis pro panoraminį langą	2021-VN-03	12	9	6			
11	Gyvūnų lentelė	2021-SK-05	12					
12	Magiškas volelis	2016-PK-03a	12					
13	Rask gyvūno vardą	2021-LT-04a		6				
14	Norai	2021-DE-08a		9	6			
15	Apelsinų sultys	2021-UY-11		9	6			
16	Vartotojo vardas	2021-HU-03		9				
17	Vandens malūnai	2021-TR-06		12	9			
18	Geriausias maršrutas	2021-CY-03		12		6		
19	Kamuolio judėjimas	2021-KR-02		12		6		
20	Ilgiausia seka	2021-UA-01b		12		6		
21	Gėlių spalvos	2016-SK-04		12				
22	Kengūra	2021-UZ-01b			6			
23	Roboto ranka	2021-DE-03			9	6		
24	Monetų maišas	2021-IE-02			9	6		
25	Ar jie susitiks?	2021-LT-06			9	6		
26	Labirintas	2021-CN-02			9			
27	Vėžlio kelias	2021-DE-07			9			
28	Braškių vagis	2021-CH-04c1			12	9	6	
29	Miško stebėjimas	2021-AT-01			12	9		
30	Taškūnai	2021-CA-01b			12		6	
31	Figūrų sukeitimas	2021-IN-05			12		6	
32	Raskite klaidą	2021-LT-07			12		6	
33	Heraklis ir hidra	2021-UZ-02			12		6	
34	Norai	2021-DE-08b				9	6	
35	Ligoninės	2021-BE-02				9		6
36	Katino nuotraukos	2021-CH-16				9		
37	Dovana	2021-IS-03				9		

Nr.	Uždavinio pavadinimas	Uždavinio identifikatorius	Nykštukai	Mažyliai	Bičiuliai	Draugai	Jauniai	Kolegos
38	Skysčiai	2021-ID-10				12	9	6
39	Reguliuojami stalai	2021-SV-01				12	9	6
40	Knyga	2021-AT-04a				12	9	
41	Skubus pasimatymas	2021-LT-01				12	9	
42	Voro antklodėlės	2021-CA-02				12	12	9
43	Gedimų nustatymas	2021-HU-02				12	12	9
44	Lakoniški pranešimai	2021-CH-06					9	6
45	Šoklusis beždžioniukas	2021-SI-02					9	6
46	Truchet'o plytelės	2021-AT-06					12	9
47	Žaidimo taisyklės	2021-CH-07a					12	9
48	Vaisių tvarka	2021-CH-13					12	9
49	Žetonų bokštai	2021-IT-01b					12	
50	Audito komitetas	2021-CZ-05						6
51	Simbolių skaitymo robotas	2021-CZ-04						9
52	Komandų būrimas	2021-CH-19						12
53	Linktelėjimų skaičiavimas	2021-DE-05						12
54	Rąstų rikiavimas	2021-DE-06						12
55	Bebrų ralis	2021-HU-04						12
56	Bebrijos susivienijimas	2021-PH-03						12
57	Du bebrai dirba	2018-LT-04						12

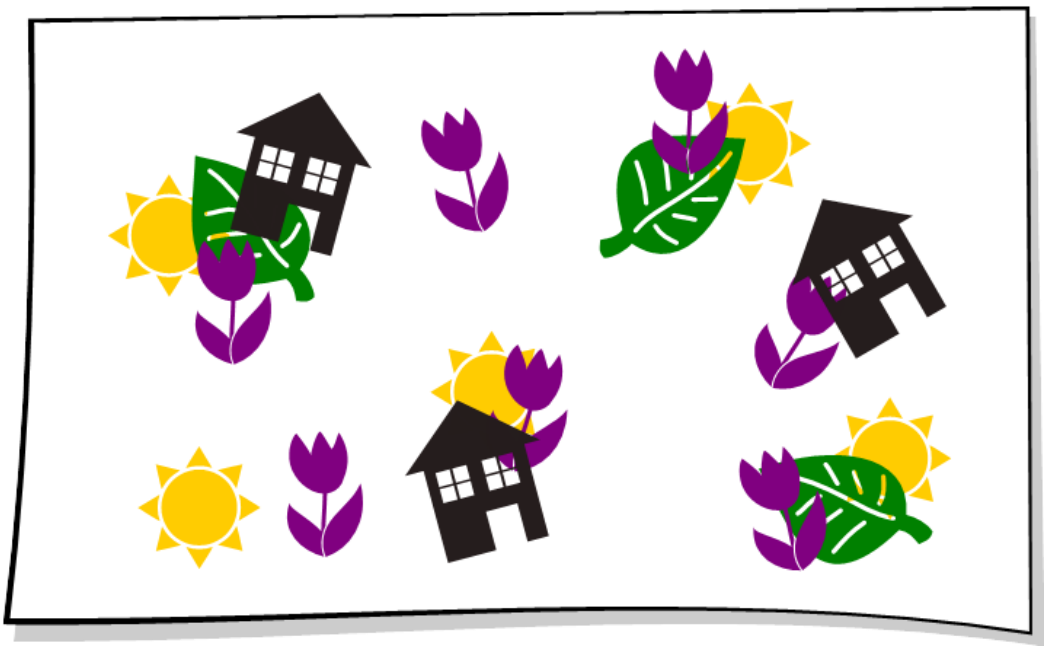
Kiekvieno uždavinio pradžioje nurodoma, kuriai amžiaus grupei jis skiriamas ir jo sudėtingumo lygis:

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

Taip pat pateikiamas uždavinio atsakymas ir paaiškinimas, kaip uždavinys susijęs su informatika.

1. Mykolo antspaudai

Mykolas turi keturis skirtingus antspaudus. Iš jų jis kuria paveikslą.



Kokia tvarka Mykolas naudoja antspaudus paveikslui sukurti?

1 2 3 4

Paaiškinimas

Teisingas atsakymas: Saulė – Lapas – Gėlė – Namas.

Pagal tai, kurios figūros uždengia kitas figūras arba yra uždengiamos kitomis figūromis, galima nustatyti, kokia tvarka Mykolas naudoja antspaudus. Saulę dengia visos kitos figūros, todėl Mykolas pirmiausia naudojo antspaudą su saule. Lapas uždengia saulę, bet jį uždengia ir gėlė, ir namas. Taigi Mykolas antruoju numeriu naudojo antspaudą su lapu. Gėlė vėl dengia ir lapą, ir saulę. Tačiau neįmanoma, kad antspaudas su gėle būtų paskutinis, nes dviejose vietose gėlę aiškiai dengia namas. Taigi Mykolas antspaudą su gėle naudojo trečią, o antspaudą su nameliu – paskutinį.

Tai informatika!

Mykolo paveikslas – tai tam tikra prasme tikrovės atvaizdas arba modeliavimas, kai figūros įspaustos keturiuose skirtinguose lygiuose: viename lygyje – saulė, kitame – lapai, trečiame – gėlės, ketvirtame – namai. Skirtingų plokštumų figūrų perdengimas leidžia Mykolui sukurti gylio ir trimačio vaizdo iliuziją ant (dvimačio) popieriaus paviršiaus. Paprastai modeliuojant vaizduojami tik tie aspektai, kurie yra būtini tam tikrai užduočiai ar funkcijai atlikti. Taigi tikrovė vaizduojama supaprastintai.

Modeliavimas – svarbi informatikos mokslo šaka, skiriami įvairūs modeliavimo tipai: objektinis, funkcinis, duomenų. Pavyzdžiui, informatikai kuria tam tikras kompiuterines programas siekdami kuo greičiau ir tiksliau ištirti realaus pasaulio sritis ir įgyti apie jas žinių. Norėdami sukurti tokią programą, informatikai pirmiausia turi sugalvoti mąstymo modelį, kuriame būtų pateikti jiems būtini realaus pasaulio elementai. Šis modelis leidžia sukurti realų modelį eskizo arba kompiuterinės programos pavidalu. Naudojant kompiuterinę programą, t. y., realaus pasaulio dalies modelį, galima įgyti žinių apie realų pasaulį.

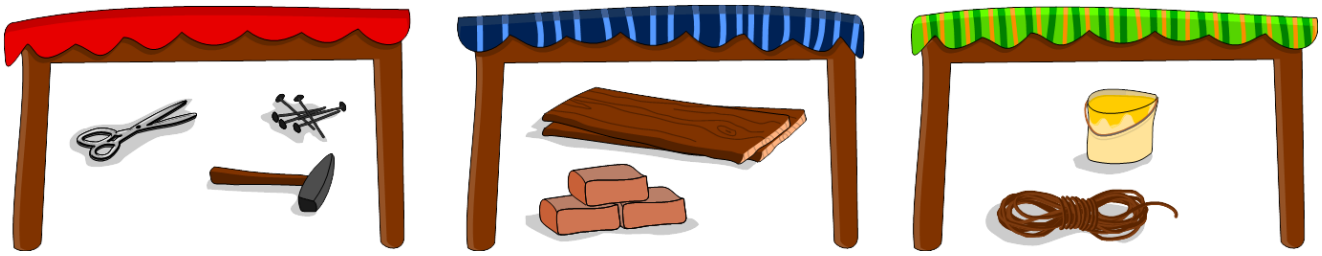
Yra daugybė kompiuterinių programų ir programėlių, kuriomis galite kurti modelius, piešti patys arba redaguoti paveikslėlius, grafiką ir nuotraukas. Daugumoje šių programų naudojami sluoksniai. Naudodami šiuos sluoksnius, be kita ko, galite nustatyti paveikslėlio ar grafinių elementų eiliškumą. Žemiausio sluoksnio paveikslėlio elementus iš dalies arba visiškai uždengia kitų sluoksnių paveikslėlio elementai. Sluoksniai gali skirtis daugeliu savybių ir funkcijų: dydžiu, spalvomis, struktūromis ar raštais, teksto ar vaizdo informacija, skaidriu ar pusiau skaidriu fonu, efektais ir dar daugiau. Animaciniuose filmuose ir kitose animacijose daug „kadrų“ (t. y., atskirų vaizdų) rodomi greitai vienas po kito, kad būtų sukurta sklandaus judėjimo iliuzija. Kiekvieno kadro apatiniame sluoksnyje paprastai yra fonas, kurį galima tiesiog nukopijuoti į kitą kadrą. Kiekvieną vaizdo sluoksnį galima redaguoti atskirai. Taip lengviau keisti, kopijuoti ar ištrinti atskiras vaizdo dalis.

2. Tilto statyba

Beata nori pastatyti tiltą per upelį. Jai reikia: plaktuko, vinių, lentų ir virvės. Rūsyje ji randa plaktuką ir virvę. Ji turi nusipirkti kitus daiktus.

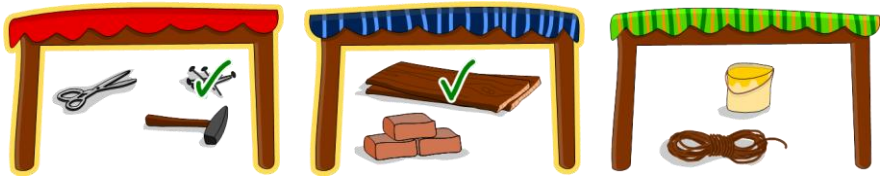
Kur Beata gali nusipirkti kitų daiktų?

Spustelėkite tinkamas parduotuves.



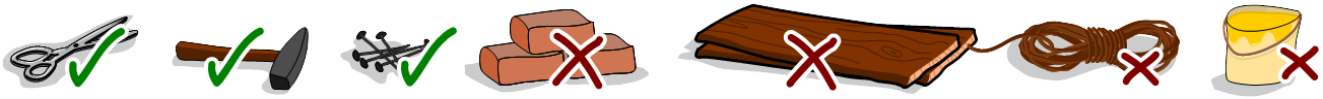
Paaiškinimas

Kairėje esančioje parduotuvėje parduodami du Beatos tiltui reikalingi daiktai: plaktukas ir vinys. Beata jau turi plaktuką, bet čia gali nusipirkti vinių. Lentų Beata gali nusipirkti viduryje esančioje parduotuvėje. Parduotuvė dešinėje parduoda virvę, tačiau Beata ją jau turi.



Todėl ji turi pasirinkti kairėje ir viduryje esančias parduotuves. Šiose parduotuvėse iš viso parduodami septyni daiktai: žirklės, plaktukas, vinys, plytos, lentos, virvė ir dažai. Tai gana daug! Du daiktai, kuriuos Beata turi nusipirkti, yra šių parduodamų prekių rinkinio dalis. Galite vaizduoti taip: parodote visus viso rinkinio daiktus ir pažymite kiekvieną daiktą, ar jis priklauso Beatos pirkimo pogrupiui, ar ne:

Taip pat galite piešti tai, ką parduoda parduotuvės – pavyzdyje parodyta, ką parduoda kairioji parduotuvė:



Iš karto galite pamatyti, ką Beata gali nusipirkti parduotuvėje kairėje: ant prekių yra žalia varnelė pirkimo pogrupyje ir pardavimo pogrupyje.

Tai informatika!

Kompiuterių programos taip pat dažnai turi lyginti kiekius. Tai galima padaryti taip: vienas bitas reikalingas kiekvienam įvykiui. Kompiuteris gali išsaugoti vieną iš dviejų bito reikšmių, pvz., „taip“ ir „ne“. Šiuo atveju saugoma, ar daiktas priklauso aibei („taip“), ar ne („ne“). Tada programa gali palyginti du rinkinius: ji patikrina, ar vieno rinkinio vieno elemento bitas yra „taip“, o kito rinkinio to paties elemento bitas taip pat yra „taip“. Tokį dviejų bitų patikrinimą kompiuteris gali atlikti ypač greitai. Todėl rinkinių su bitais aprašymas yra labai paplitęs kompiuterių moksle.

3. Monetų dėlionė

Emilija turi 6 monetas:

Emilija monetas sudėliojo taip, kaip parodyta paveikslėlyje.

Kurią monetą ji padėjo ketvirtą? Spustelėk tą monetą.

Paaiškinimas

Teisingas atsakymas:  .

Monetos buvo sudėtos tam tikra tvarka. Visos monetos yra bent kažkiek uždėtos ant kitos monetos, taigi atsakymo galima pradėti ieškoti nuo pačios viršutinės monetos. Pati viršutinė moneta, kuri neuždengia nė vienos kitos monetos, yra  , tolesnė moneta, kurią dengia tik  , yra  . Ketvirta moneta negali būti uždengta niekuo daugiau tik  ir  monetomis, todėl ta moneta yra  .

Tai informatika!

Monetos išdėliotos tam tikra tvarka. Tą patį galima matyti piešiant ką nors kompiuteriu: jei nupiešite skritulį, ant jo du taškus ir kreivą liniją, gausite veiduką. Jei būtumėte skritulį nupiešę paskutinį, dviejų taškų ir linijos nesimatytų.

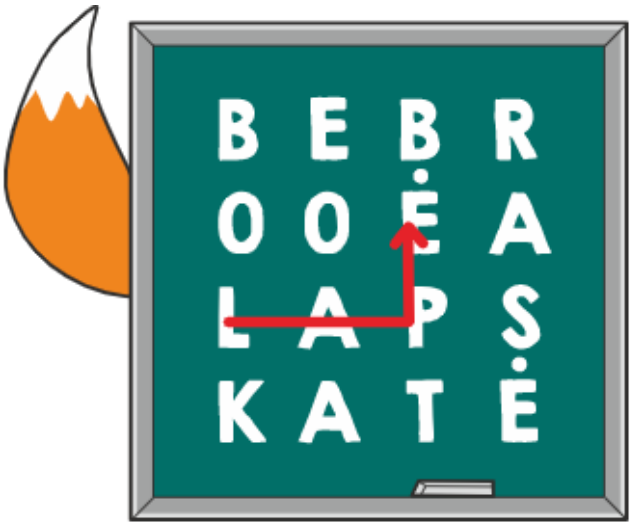
Kompiuteriai taip pat dirba pagal tam tikrą tvarką. Daugelis kompiuterinių programų yra parašytos taip, kad įvyksta vienas veiksmas ir tik tada kitas. Štai taip atrodo kompiuterinė programa, kuri turi nupiešti veiduką:

```
Piešti apskritimą koordinatėse (5,5), spindulys 5
Piešti tašką koordinatėse (2,7)
Piešti tašką koordinatėse (7,7)
Piešti liniją, pakreiptą į kairę nuo (2,2) iki (7,2)
```

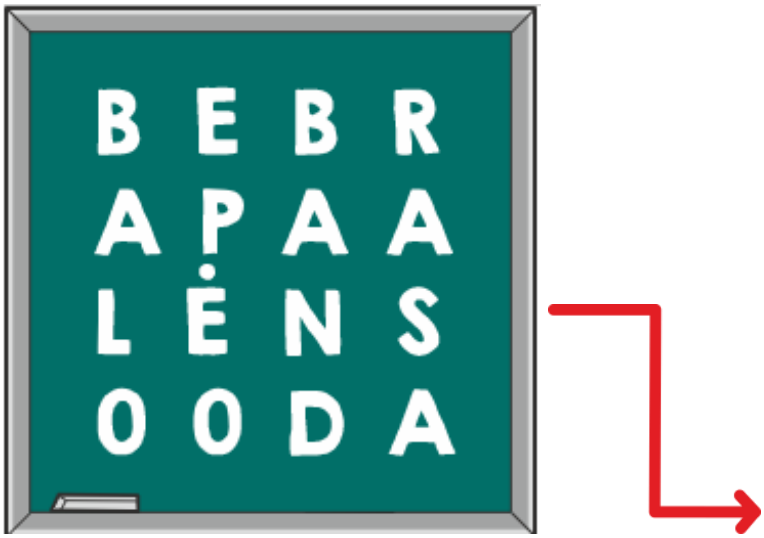
Ne visos kompiuterių programos veikia šitaip. Sudėtingesnėms programoms reikia naudoti veiksmų pasirinkimo ir kartojimo sąlygas.

4. Rask gyvūno vardą

Sekdami pateikta rodykle raidžių lentelėje galime perskaityti gyvūno vardą. Pavyzdžiui, tokią rodyklę uždėję ant paveiksliuke parodytos raidžių lentelės perskaitome žodį LAPĖ.



Pritaikykite žemiau pateiktą rodyklę raidžių lentelei ir perskaitykite gyvūno vardą.



Paaiškinimas

Atsakymas: PANDA.

Turime taikyti laužtinės rodyklės konfigūraciją raidžių lentelei ir skaityti įvairius žodžius. Šios laužtinės rodyklės konfigūraciją galime aprašyti taip: 2 langeliai į dešinę, 3 žemyn, vėl 2 į dešinę. Pirmoje eilutėje galime tikrinti kai kuriuos atvejus (kad ir BE ar net BEB, bet BEBR yra tikrai per ilgas). Antroje eilutėje greit aptiksime tinkamą derinį PA ir tada žemyn ND – lengva, randame žodį PANDA.

Tai informatika!

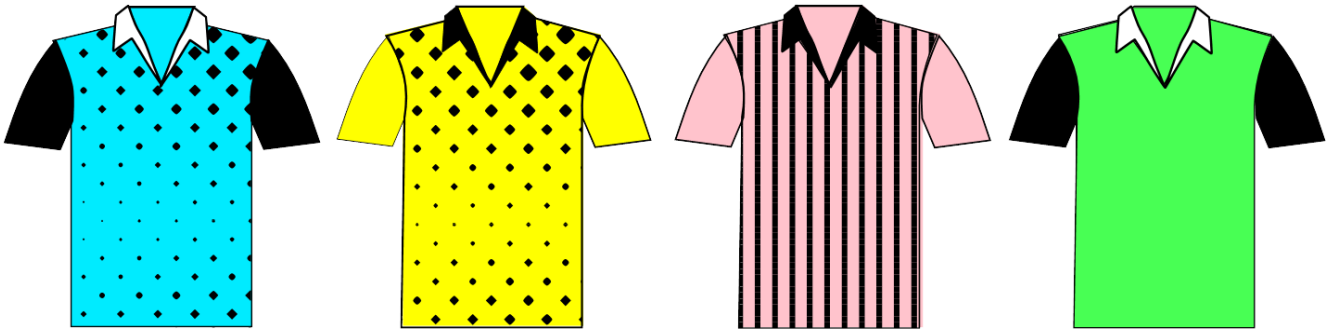
Uždavinio esmė – ieškojimas dvimačiame raidžių lauke (dvimatis masyvas) taikant laužtinės rodyklės schemą, iš tiesų – vadinamąjį šabloną.

Šablonas dažnai naudojamas informatikoje, ypač programavime.

5. Futbolo marškinėliai

Ana kraunasi krepšį futbolo rungtynėms. Jai reikia pasiimti nedryžuotus marškinėlius juoda apykakle ir ne juodomis rankovėmis.

Kuriuos marškinėlius Ana turėtų pasiimti?



A. B. C. D.

Paaiškinimas

Teisingas atsakymas: B.

A marškinėliai netinka, nes jų rankovės juodos.

B marškinėliai tinka, nes jų rankovės nėra juodos (geltonos), apykaklė juoda ir jie nedryžuoti.

C marškinėliai netinka, nes yra dryžuoti.

D marškinėliai netinka, nes jų rankovės juodos.

Tai informatika!

Šioje užduotyje turi būti tenkinamos trys dalinės sąlygos (rankovių spalva, apykaklės spalva, dryžuotumas). Suprasti sąlygų naudojimą yra labai svarbu programuojant – sąlygas galima sutikti visose programavimo kalbose! Jomis galima nusakyti, kurios programos dalys turėtų būti vykdomos toliau.

Ši užduotis gali būti naudojama pristatyti logines operacijas IR ir NE (AND ir NOT). Jei keletas dalinių sąlygų turi būti tenkinamos kartu, naudojame IR, pvz., rankovių spalva = šviesi IR apykaklės spalva = juoda. Jei dalinė sąlyga turėtų būti netenkinama, naudojame NE, pvz.: NE (marškinėlių ornamentas = dryžiai).

6. Raidės

Lina ir Ana naudoja slaptą susirašinėjimo sistemą bendrauti tarpusavyje. Ana parašė žinutę ir paklausė Linos, kieno sostinė yra Ryga.

Lina atsakė LATVIJOS, bet atsakymą užrašė šitaip:

Vėliau Lina paklausė Anos, ką mokytoja uždavė namų darbams – perskaityti knygą ar pažiūrėti televizijos laidą.

Kaip Ana užkoduotų atsakymą, jei šis būtų TV?

Paaiškinimas

T raidė yra trečioji žodyje LATVIJOS. V raidė yra ketvirtoji tame žodyje. Atitinkamai randami šias raides žymintys jaustukų dvejetai.

Tai informatika!

Kriptografija – informacijos pateikimo būdas naudojant šifravimą, kad informaciją galėtų perskaityti tik tie, kam ji ir yra skirta. (Graikų kalba „kryptós“ reiškia „paslėptas“, o „graphein“ – „rašyti“.)

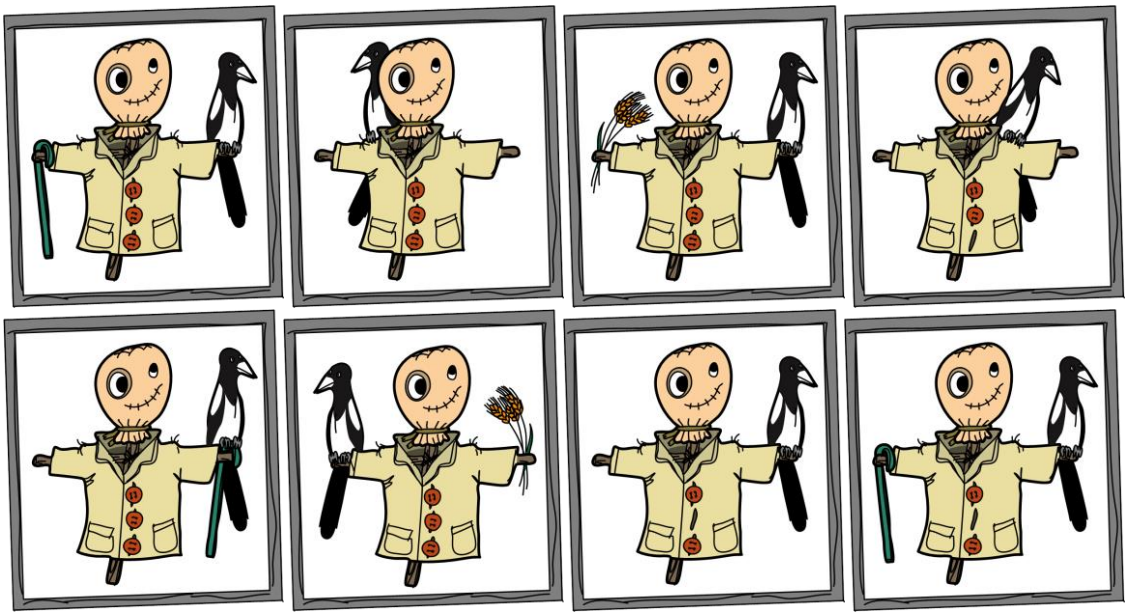
Kompiuterijoje kriptografija naudojama labai dažnai, yra daug kriptografijos metodų. Viena iš sričių, kur naudojama kriptografija – kompiuterinio saugumo sistemos.

Šios užduoties kodų lentelė panaši į dvejetainę kodų lentelę, tik vietoje skaitmenų 0 ir 1 raidėms užkoduoti naudojami jaustukai.

7. Kaliausė

Saulenė nupiešė kelias kaliauses. Jos mėgstamiausia kaliausė:

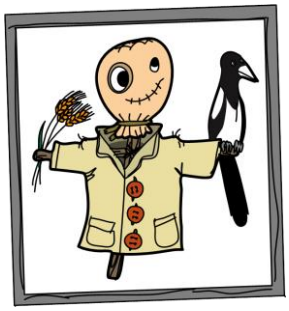
- užsegtomis visomis palto sagomis;
- šarka tupi ant kairės rankos;
- nelaiko lazdelės.



Paspausk ant jos mėgstamiausios kaliausės.

Paaiškinimas

Teisingas atsakymas: trečioji kaliausė:



1, 2, 3, 5 ir 6 piešiniuose esančios kaliausės atitinka pirmąją sąlygą (užsegtos visos palto sagos).

Piešiniai 1, 3, 5, 7 ir 8 atitinka antrąją sąlygą (šarka tupi ant kairės rankos).

Piešiniai 2, 3, 4, 6 ir 7 atitinka trečiąją sąlygą (nelaiko lazdelės).

Taigi vienintelis trečiasis piešinys atitinka visas tris sąlygas.

Tai informatika!

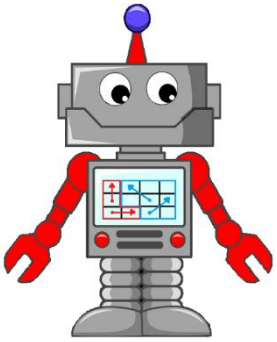
Tai logikos algebros uždavinys. Sprendimui turime pritaikyti vieną iš loginių operacijų – sujungimą (IR). Operacijos sekų sąjunga yra teisinga tik tokiu atveju, jei visos operacijos yra teisingos.

Šitaip naudodami 0 (neteisinga) ir 1 (teisinga) galime užkoduoti visus atsakymus:

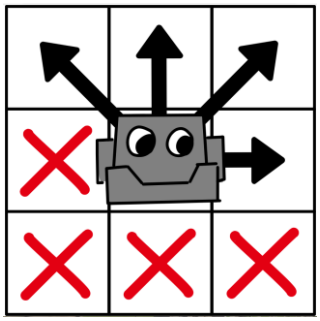
	1	2	3	4	5	6	7	8
1 sąlyga	1	1	1	0	1	1	0	0
2 sąlyga	1	0	1	0	1	0	1	1
3 sąlyga	0	1	1	1	0	1	1	0

Matome, kad tik trečiojo stulpelio visi atsakymai vienetai – teisinga (1,1,1).

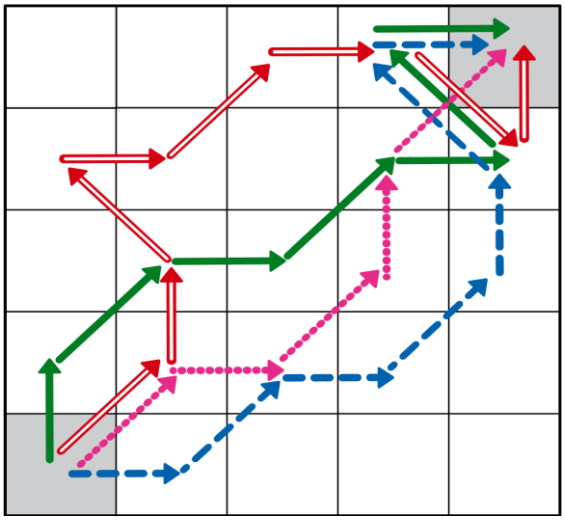
8. Roboto kelias



Robotukas nori pasiekti priešingą lauko kampą. Jis gali žengti tik 4 kryptimis:



Ir dar – robotukas negali atlikti dviejų tos pačios krypties žingsnių iš eilės.



Kuriuo keliu eidamas robotas negalėtų pasiekti tikslo?

Paaiškinimas

Kelias, pažymėtas raudonomis dviejų linijų strėlėmis, yra blogas. Robotas šiuo keliu eiti negalės, nes naudojamas žingsnis įstrižai žemyn dešinėn, o toks žingsnis nėra leistinas.

Tai informatika!

Šioje užduotyje pateikiamas algoritmas, kuris susideda iš galimų žingsnių sekos. Tačiau yra ir apribojimų: galima žengti žingsnius tik 4 kryptimis ir tik taip, jog du iš eilės einantys žingsniai nebūtų ta pačia kryptimi. Tai reiškia, jog reikės galvoti algoritmiškai, įtraukiant žinomas taisykles, ir tokiu būdu rasti netinkamą kelią.

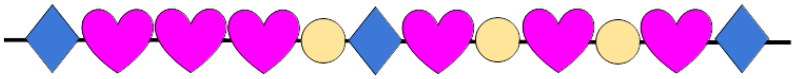
9. Vėrinys

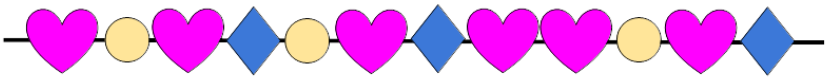
Aiva, Diana ir Vida pasidarė vėrinius su savo vardais. Raidėms jos naudoja dviejų rūšių karoliukus:  ir  . Atskiras raides jos atskiria  karoliuku.

AIVA 

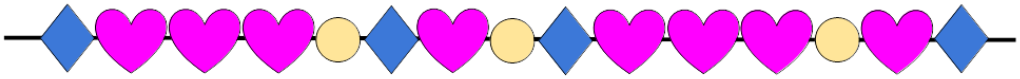
DIANA 

Kokį vėrinį pasidarė Vida?

A. 

B. 

C. 

D. 

Paaiškinimas

Teisingas atsakymas: A.

Mūsų ieškomo vardo pirma raidė V yra varde AIVA. Ji pavaizduota šia karoliukų seka:

Žinant, kad mūsų ieškomo vardo pirma raidė taip pat yra V, galime surasti visus vėrinius, kurie prasideda šia karoliukų seka. Lieka du vėriniai A ir D. Pažiūrėję į AIVOS vėrinį matome, kad raidė I yra atvaizduota  .

Pridėjus antrą raidę abu variantai A ir D vis dar yra teisingi. Toliau matome, kad mums reikia surasti karoliukus kurie atvaizduoja raidę D, pagal Dianos vėrinį galime sužinoti, kad raidė D yra atvaizduota karoliuku  . Sudėję visus šiuos karoliukus matome, kad lieka tik vienas variantas, A.

Kitas sprendimo būdas būtų paimti Dianos ir Aivos vėrinius ir kiekvienai raidei priskirti po simbolių seką. Tuomet pagal tai iššifruoti visus atsakymo variantus. Tokiu būdu mes gausime, kad duoti atsakymo variantai yra tokie: A) VIDA, B) DIVA, C) AIDA, D) VIVA. Iš čia matysime, kad teisingas atsakymas yra A.

Tai informatika!

Dažnai koduojame informacija, kad būtų lengviau ir ekonomiškiau komunikuoti tarpusavyje. Šiuo atveju informacija yra koduojama slaptu kodu, kur kiekviena raidė atitinka tam tikrą simbolių kombinaciją.

Jei nežinome, koks kodavimas yra naudojamas, tai negalime sukurti vėrinių kitokiems vardams. Mums reikėtų nuspręsti kaip reikia koduoti kiekvieną abėcėlės raidę. Kodavimas neapsiriboja teksto kodavimu, taip pat galima koduoti nuotraukas, garsus, vaizdo įrašus.

10. Žvilgsnis pro panoraminį langą

Keturi bebriukai Agnė, Benas, Cezaris ir Dalė iš pažymėtų vietų 1, 2, 3 ir 4 žiūri į keturis medžius A, B, C ir D.

Bebriukai išvardija, kokia tvarka jie mato medžius, kai žiūri iš kairės į dešinę.

Agnė: D-A-B-C

Benas: C-D-A-B

Cezaris: C-A-D-B

Dalė: C-D-A-B

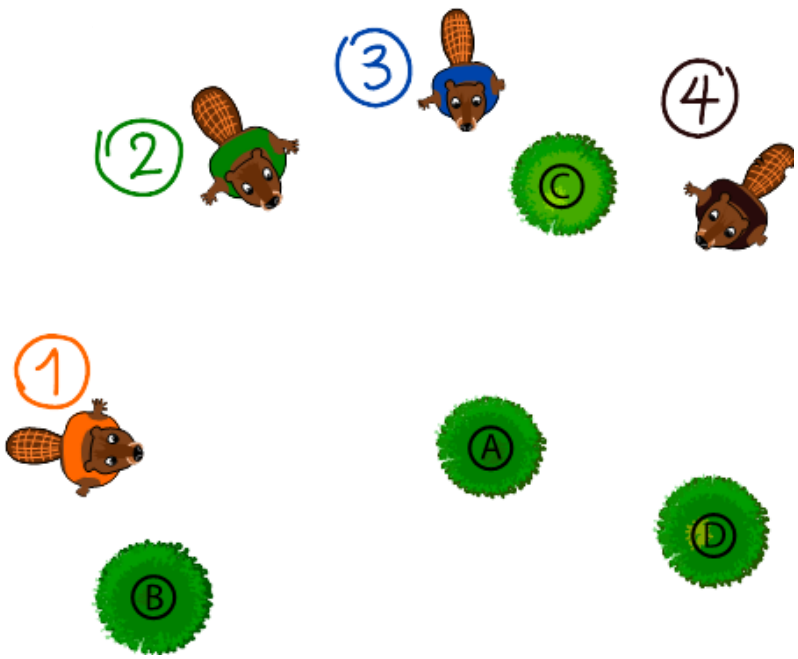
Kuris bebriukas yra 1-oje vietoje?

A. Agnė

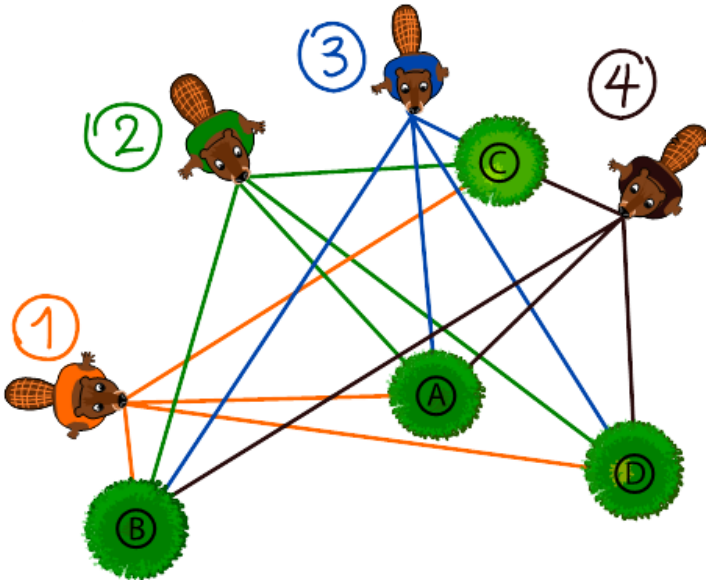
B. Benas

C. Cezaris

D. Dalė



Paaiškinimas



Teisingas atsakymas: C (Cezaris).

Paaiškinti geriausia brėžiant linijas iš kiekvieno bebriuko vietos į kiekvieną medį.

Skirtingiems bebriukams pasirenkama skirtinga linijų spalva. Kiekvieną bebriuką su medžiais jungiame linijomis pradėdami nuo kairiausiai matomo medžio ir judėdami į dešinę.

1 vieta: C-A-D-B
2 vieta: C-D-A-B
3 vieta: C-D-A-B
4 vieta: D-A-B-C

Palyginę gautus užrašus su tuo, ką bebriukai pasakė, matome, kad 1-oje vietoje yra Cezaris.

Tai informatika!

Duomenų struktūros yra duomenų organizavimo, valdymo ir saugojimo būdas, kuris leidžia efektyviai pasiekti duomenis ir juos keisti. Duomenų struktūrų yra įvairių tipų ir jos pritaikytos įvairioms programoms. Kai kurios duomenų struktūros yra specializuotos konkrečioms užduotims atlikti. Šioje užduotyje naudojama abstrakti duomenų struktūra – grafas. Grafų teorija lyra atskira matematikos šaka, ji tyrinėja grafą kaip matematinį objektą, sudarytą iš viršūnių ir briaunų. Grafai dažnai naudojami įvairiems praktiniams uždaviniams spręsti.

Į šią užduotį galima pažvelgti ir kiek kitaip. Kompiuteriai buvo išrasti sudėtingiems skaičiavimams atlikti. Vėliau paaiškėjo, kad jie gali apdoroti bet kokią informaciją, kuri suskaitmeninta, t. y., užkoduota dvejetainiais skaičiais. Kompiuteriai apdoroja ir vaizdinę informaciją, pavyzdžiui, fotoaparatų nuotraukas ar vaizdo įrašus. Norint gauti informacijos iš vaizdinių duomenų, reikia sukurti algoritmus, kurie būtų susiję su duomenų geometrine informacija.

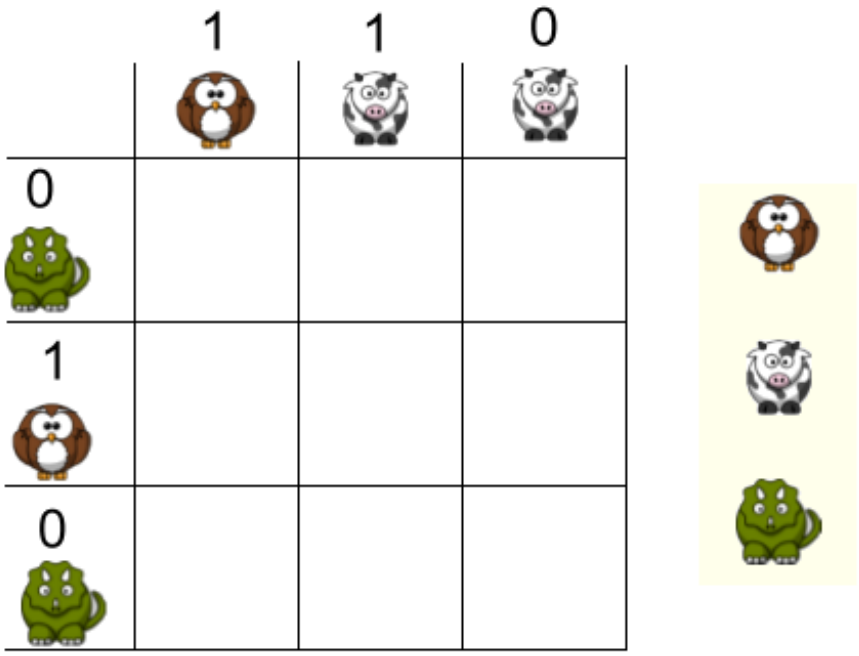
Atlikdami šią užduotį naudojame geometrinę informaciją. Kiekvieno bebriuko linijos nuo jų padėties iki medžių turi skirtingas kryptis. Šias kryptis galime apibūdinti kampais. Iš to, ką sako bebriukai, negalime sužinoti tikslių kampų, bet žinome jų eiliškumą, tai yra, kuris kampas yra mažesnis ar didesnis už kitą. Dažnai tokie, atrodytų, paprasti duomenys padeda atlikti sudėtingas užduotis, pavyzdžiui, identifikuoti asmenis.

11. Gyvūnų lentelė

Lentelė sudaryta iš 9 langelių, išdėstytų trimis eilutėmis ir trimis stulpeliais. Virš stulpelių ir eilučių pradžioje nupiešti gyvūnai. Skaičius prie gyvūno nurodo, kiek tokių gyvūnų turi būti atitinkamoje eilutėje ar stulpelyje.

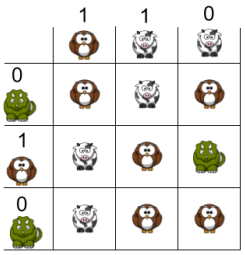
Pavyzdžiui, paskutiniame stulpelyje prie karvės yra 0, tad jame gali būti tik kitų dviejų rūšių gyvūnai.

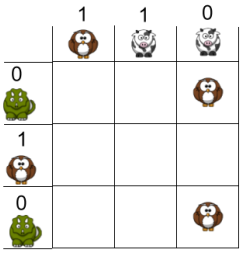
Nuvilk gyvūnų paveikslėlius ir užpildyk visus lentelės langelius taip, kad gyvūnų skaičiai ir eilutėse, ir stulpeliuose atitiktų nurodytus.



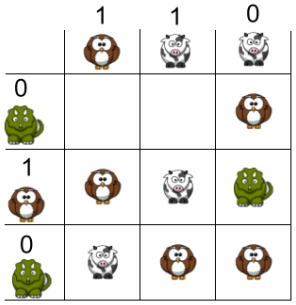
Paaiškinimas

Vienas iš teisingų sprendimų pavaizduotas paveiksle. Spręsdami uždavinį interaktyviai, pirmiausia užpildome langelius, kuriuose tikrai žinome, koks turi būti gyvūnas – pirmojoje eilutėje ir paskutiniame stulpelyje negali būti nei dinozauro, nei karvės, tad ten bus pelėda. Tą patį galime pasakyti ir apie trečiąją eilutę, ir apie trečiąjį stulpelį.





Daugiau nėra vienareikšmiškai aišku, koks gyvūnas kokiame langelyje turėtų būti.



Sprendimo ieškosime bandymų metodu: į langelius po vieną dėsime gyvūnus taip, kad nurodyti skaičiai išliktų teisingi ir eilutėms, ir stulpeliams (nors ir laikinai). Šitaip bandydami galime pasiekti tokią situaciją, kad kažkuriuo metu nebegalėsime pildyti lentelės. Tada teks grįžti ir bandyti dėti kitą gyvūną. Vis dėlto lentelę užpildyti pagal nurodytus skaičius nėra sunku.

Yra dar 22 sprendimo variantai, kuriuos galima rasti parašius „Python“ kalba skriptą. Štai visi 23 teisingi sprendimo variantai:

PPP KPD KKP	PPP KKP KPP	PPP KDP KKP	KPP KPD PKP	KPP KKP PPP	KPP KDP PKP
PPP DPD KKP	PPP DKP KPP	PPP DDP KKP	KPP DPD PKP	KPP DKP PPP	KPP DDP PKP
PKP KPD KPP	PKP KDP KPP	PKP DPD KPP	KKP PDD KPP	KKP KPD PPP	KKP KDP PPP
PKP DDP KPP	KPP PKD KPP	KPP PDD KKP	KKP DPD PPP	KKP DDP PPP	

Tai informatika!

Vienas iš uždavinių sprendimo metodų, aprašytas matematiko George'o Polya – tai spėjimo ir tikrinimo metodas.

Informatikoje tokie uždaviniai vadinami „tenkinimo su ribojimais uždaviniais“. Sprendžiant tokius uždavinius siekiama sukurti tokią situaciją, kad aibė kintamųjų (visi gyvūnai) atitiktų tam tikrų taisyklių (ribojimų) rinkinį (neleidžiama dėti gyvūno paveikslėlio į kai kuriuos langelius).

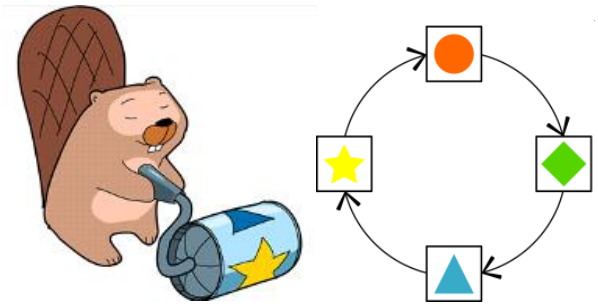
Aibė objektų, šiuo atveju gyvūnų, turi tenkinti tam tikrus ribojimus, nurodytus lentelės eilučių ir stulpelių aprašuose. Kintamųjų (gyvūnų paveikslėlių) ribojimų šiame uždavinyje yra labai mažai. Ribojimų tenkinimo uždaviniai tyrinėjami dirbtinio intelekto srityje – ten jie būna labai sudėtingi.

Uždavinys sprendžiamas ieškant langelių, kuriuose vienareikšmiškai aišku, koks gyvūnas turi būti.

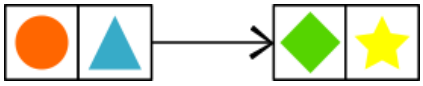
Visiems uždavinio sprendiniams rasti reikia naudoti grįžimo metodą. Juo bandoma numatyti kitą galimą žingsnį, kad sprendimas vis dar būtų teisingas. Jei sprendimas pasirodo neteisingas, taikant grįžimo metodą grįžtama atgal prie paskiausio teisingo žingsnio ir ieškoma kito dar nebandyto tolesnio žingsnio.

12. Magiškas volelis

Bebras dažo figūrėles magišku voleliu. Jis gali keisti vieną figūrėlę kita pagal schemos rodykles.



Štai ką bebras gauna iš skritulio ir trikampio:



Bebras taiko magišką volelį šiai figūrėlių eilei:

Kaip atrodys gauta figūrėlių eilė? Nuvilkite reikiamas figūrėles į langelius.

Paaiškinimas

Teisingas atsakymas:

Atsakymas gaunamas pritaikius užduotyje aprašytą algoritmą.

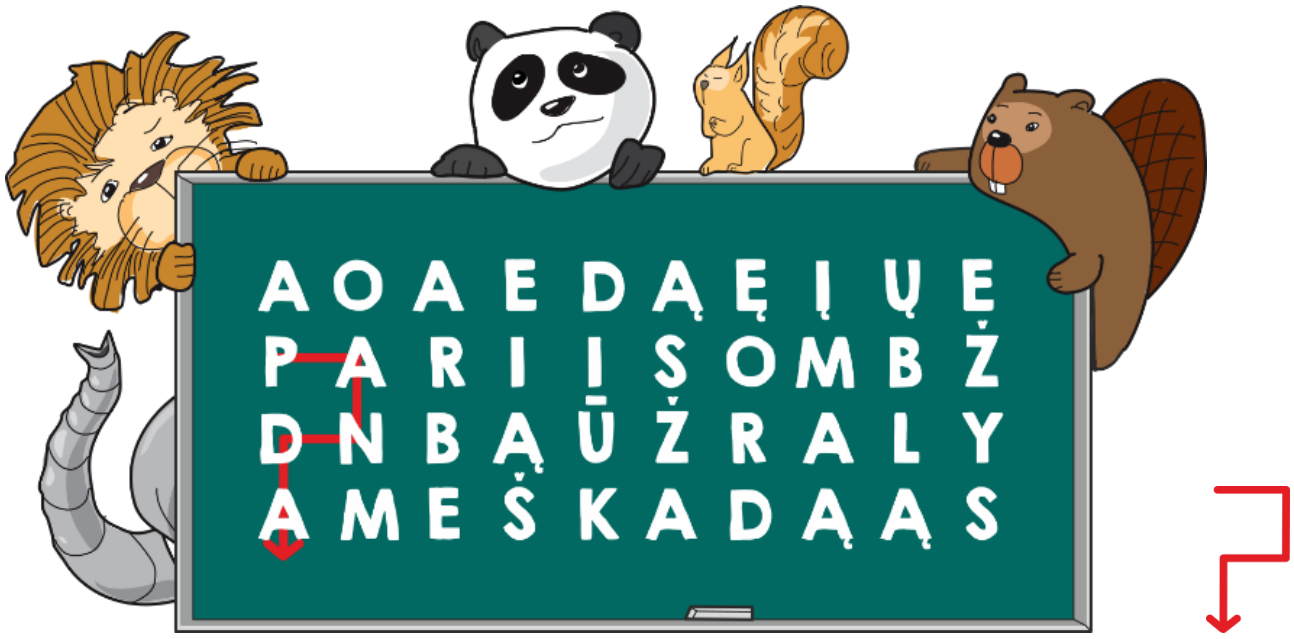
Tai informatika!

Ši užduotis supažindina su algoritmo sąvoka. Algoritmas – tai veiksmų seka, kurią reikia atlikti, kad gautum norimą rezultatą. Aprašytas algoritmas yra supaprastintas kompiuterinės regos algoritmas, kai pikseliai keičiami remiantis jų reikšmėmis. Pavyzdžiui, kai norima pakeisti paveikslėlio ryškumą, pritaikomas koks nors filtras arba atliekamas kitoks paveikslėlio keitimas.

Taip pat šiuo uždaviniu mokoma skaityti schemas, kaip viena figūrėlė gali būti keičiama kita.

13. Rask gyvūno vardą

Sekdami pateikta laužtine rodykle raidžių lentelėje galime perskaityti gyvūno vardą. Pavyzdžiui, greta šio paveiksliuko pateiktą rodyklę uždėję ant raidžių lentelės perskaitome žodį PANDA:



Pritaikykite žemiau pateiktą laužtinę rodyklę raidžių lentelei ir perskaitykite gyvūno vardą:

Paaiškinimas

Teisingas atsakymas: DRAMBLYS.

Turime taikyti laužtinės rodyklės konfigūraciją raidžių lentelei ir skaityti įvairius žodžius. Galime įžvelgti, kad rodyklės konfigūracija yra simetriška: 2 raides jungianti atkarpa aukštyn, tada 2 raides jungianti atkarpa į dešinę, 2 raides aukštyn, 2 raides į dešinę, tada 2 raides žemyn, 2 raides į dešinę ir galiausiai 2 raides jungianti atkarpa žemyn. Žodis gali prasidėti paskutinėje arba priešpaskutinėje eilutėje. Imkime priešpaskutinę eilutę. Tikriname visas raidžių poras: DP, NA, BR, ĄI, ŪI, ŽS, RO – toliau nebereikia tikrinti, netilps. Įsitikiname, kad radome vieną žodį – BRIEDIS, deja, jis per trumpas, nesimetriškai baigtųsi.

Analogiškai tikriname paskutinę eilutę ir randame žodį DRAMBLYS.

Tai informatika!

Uždavinio esmė – ieškojimas dvimačiame raidžių lauke (dvimatis masyvas) taikant laužtinės rodyklės schemą, iš tiesų – vadinamąjį šabloną. Šablonas dažnai naudojamas informatikoje, ypač programavime.

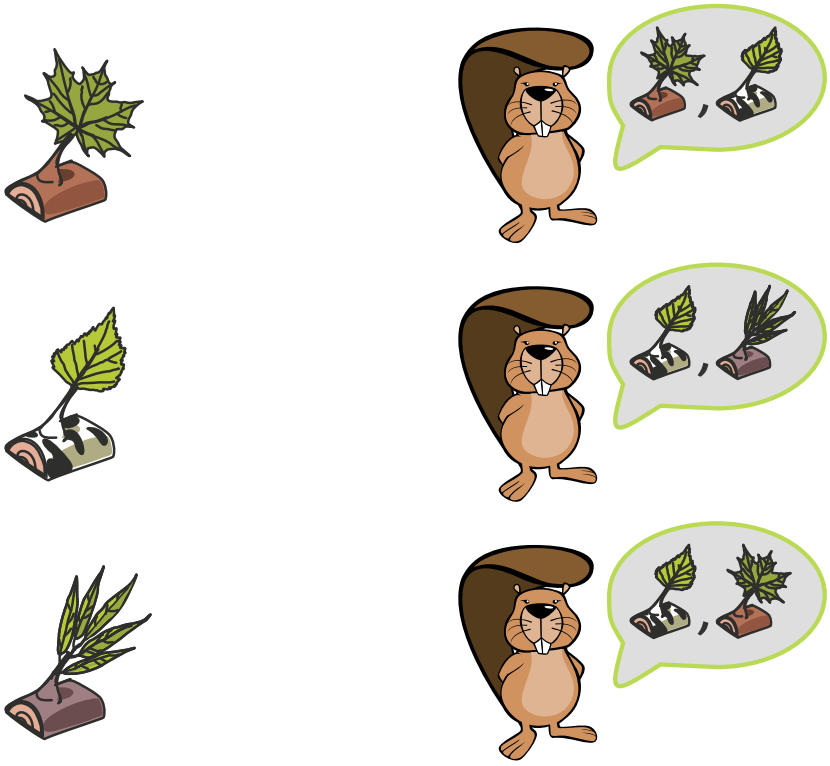
14. Norai

Bebrų šeima turi tris dovanas savo vaikams. Kiekvienas vaikas yra išsirinkęs labiausiai pageidaujamą dovaną arba, jei jos negautų, kitą.

Tėvai bebrai nori paskirstyti dovanas taip, kad:

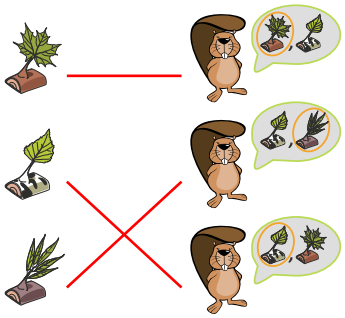
- kuo daugiau vaikų gautų labiausiai norimą dovaną,
- kiti gautų antrą pagal pageidavimą dovaną.

Įteik dovanas. Nuvilk dovanas prie bebriukų.



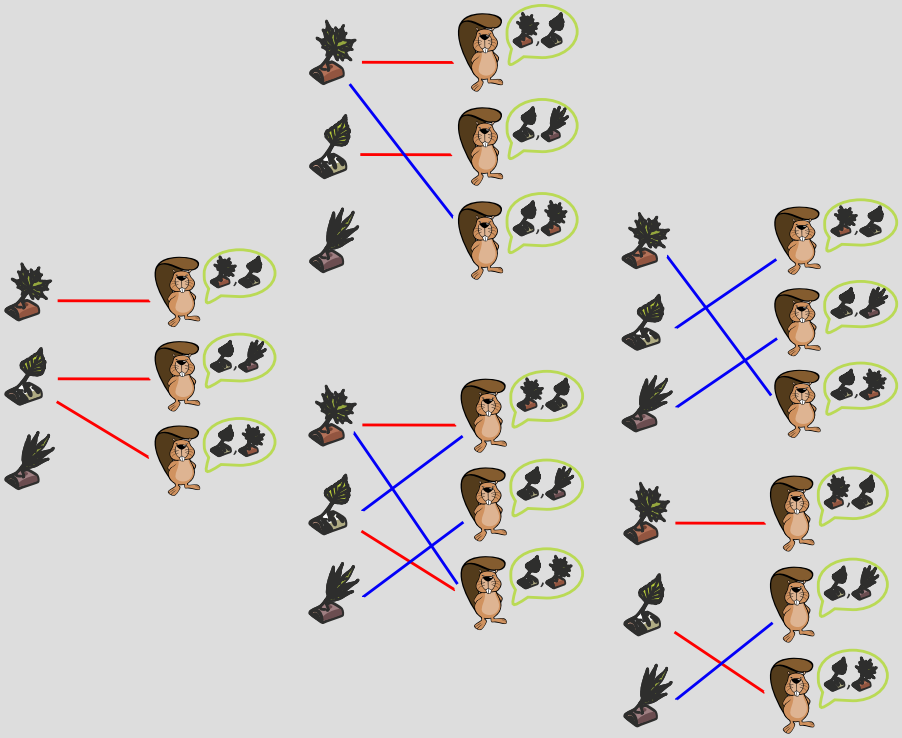
Paaiškinimas

Paveiksle pavaizduotas dovanų išdalinimas. Pradžiuginti visus bebriukus jų labiausiai norima dovana neįmanoma, nes du bebriukai labai nori tos pačios dovanos. Parodytame atsakymo paveiksle du bebriukai gauna jų labiausiai norimas dovanas, o vienas bebriukas gauna antrą pagal pageidavimą dovaną. Šiai užduočiai geresnis scenarijus neįmanomas. Uždavinį sprendžiant iš viršaus į apačią, antra dovana yra priskiriama antram bebriukui (kaip ir pirma dovana priskiriama pirmam bebriukui), tad trečias bebriukas nebegali gauti labiausiai geidžiamos dovanos. Tai yra vienintelis galimas sprendimas.



Tai informatika!

Kokie algoritmai naudojami sprendžiant tokius uždavinius? Informatikos specialistams dažnai tenka susidurti su išvardijamų numeravimų uždaviniu. Mūsų atveju pirmiausia reikia apsvarstyti tas dovanas, kurių bebriukai labiausiai nori. Paveiksle jos pažymėtos raudonai. Matome, kad du bebriukai nori tos pačios dovanos.



Kadangi tik vienas iš šių bebriukų gali gauti labiausiai pageidaujamą dovaną, turime rinktis iš dviejų galimų variantų:

1) antras bebriukas gauna antrą dovaną arba
2) antras bebriukas negauna antros dovanos.

Abiem šiems variantams pateiktos iliustracijos, kur dovanos, kurias bebriukai nori gauti antru lygiu, yra pažymėtos mėlynai. Pirmame pavyzdyje mėlyni jungikliai gali būti pašalinti nuo antro bebriuko ir antros dovanos, kadangi antra dovana jau buvo priskirta antram bebriukui. Antrame pavyzdyje vis dar yra daug galimybių, bet galime daryti tą patį, ką darėme prieš tai – padalinti dovanas taip, kad trečias bebriukas gautų pirmą dovaną arba jis jos negautų. Taip darydami mes galime gauti visus įmanomus variantus ir išsirinkti geriausią. Šis algoritmas yra vadinamas „skaldyk ir valdyk“ algoritmu.

15. Apelsinų sultys

Bebrai žaidžia loginį žaidimą su apelsinų sultimis.

Bebras Jonas gali gerti iš butelio:

a) jei pasirinkto butelio kairėje gretimame butelyje yra mažiau sulčių, ir

b) jei pasirinkto butelio dešinėje gretimame butelyje yra daugiau sulčių.

Iš kelių butelių Jonas gali gerti sultis?

A. 3

B. 2

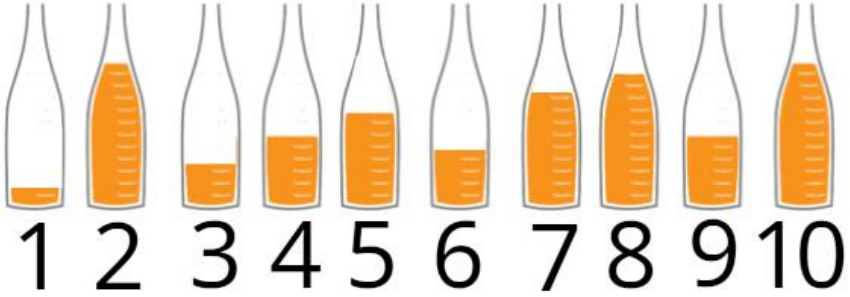
C. 5

D. 6

Paaiškinimas

Teisingas atsakymas: B (iš dviejų butelių).

Tik ketvirtas ir septintas buteliai atitinka duotus reikalavimus (mažiau sulčių kairėje IR daugiau sulčių dešinėje).



Tai informatika!

Kompiuterių mokslas kartais reikalauja spręsti uždavinius, kur yra ribojimai, apibrėžti loginėmis sąlygomis. Reikia rasti sprendimą, tenkinantį visus reikalavimus. Sudėtingesnė užduotis, kai loginiai reikalavimai jungiami naudojantis loginėmis operacijomis. Pavyzdžiui, konjunkcija (abu A ir B reikalavimai turi būti tenkinami) arba disjunkcija (kai užtenka, kad vienas reikalavimas būtų tenkinamas).

Procesas, kurį turi atlikti bebras, yra labai panašus į planavimo metodą. Ši užduotis taip pat skatina suprasti algoritmą ir atpažinti, kas bus toliau. Remiantis duota situacija ir algoritmu reikia pažingsniui išspręsti uždavinį – būtent tas darbas, kurį atlieka programuotojai. Toks proceso stebėjimas vadinamas derinimu.

16. Vartotojo vardas

Kai naujas mokinys priimamas į mokyklą, jam suteikiamas kompiuterių laboratorijos vartotojo vardas. Jis sudaromas taip: paimamos pirmosios trys mokinio pavardės raidės, po jų eina metų, kada mokinys priimtas į mokyklą, paskutiniai trys skaitmenys.

Vakar į mokyklą atėjo keturi nauji mokiniai:

Vardas	Pavardė
Paulius	Pinkus
Karolis	Juška
Gerda	Petrauskaitė
Ona	Kairytė

Šiandien atėjo dar viena mokinė – Karolina Petraitytė. Matome, kad jos vartotojo vardas bus toks pat, kaip ir kito mokinio.

Kas turi tą patį vartotojo vardą, kaip ir Karolina Petraitytė?

A) Paulius Pinkus

B) Karolis Juška

C) Gerda Petrauskaitė

D) Ona Kairytė

Paaiškinimas

Teisingas atsakymas: C – Gerda Petrauskaitė.

Karolinos Petraitytės vartotojo vardas yra „pet021“. Pirmos trys Petraitytės pavardės raidės yra „pet“. Ji priimta į mokyklą 2021 metais, taigi paskutiniai trys metų skaitmenys yra 0, 2 ir 1. Sujungus gauname vartotojo vardą „pet021“. Kitų mokinių vartotojų vardus galime sukurti taip pat. Gauname:

Mokinys	*Vartotojo vardas*
Paulius Pinkus	pin021
Karolis Juška	juš021
Gerda Petrauskaitė	pet021
Ona Kairytė	kai201

Kaip matome, Karolina Petraitytė ir Gerda Petrauskaitė turi tą patį vartotojo vardą.

Tai informatika!

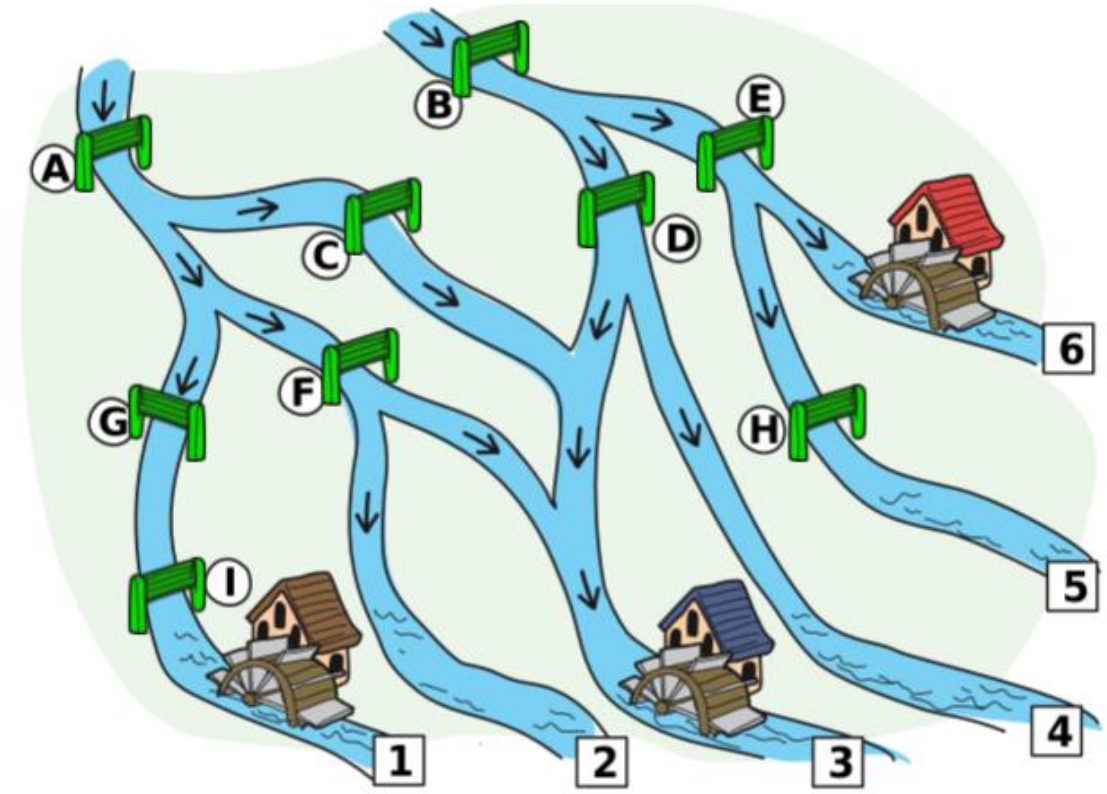
Kai kompiuterį naudoja keli žmonės, svarbu, kad jie visi prisijungtų naudodami skirtingą vartotojo vardą. Kitaip vieno asmens dokumentus gali perskaityti ar pakeisti kitas asmuo, turintis tą patį vartotojo vardą.

Sistema, kurią naudoja mokykla, nėra labai gera, nes mokiniams, turintiems panašią pavardę, dažnai bus suteiktas tas pats vartotojo vardas. Tiesą sakant, vartotojo vardai kuriami naudojant sudėtingesnes taisykles (pavyzdžiui, naudojant ir vardą, ir pavardę), tačiau net ir tada gali atsitikti taip, kad du asmenys gaus tą patį vartotojo vardą. Dažnai papildoma skaičiais, kad vardai būtų skirtingi.

Tiesa, nėra saugu vartotojo varde (arba slaptažodyje) nurodyti gimimo datą. Tai gali sumažinti privatumą.

17. Vandens malūnai

Ūkininkas Majus turi tris malūnus. Majus nori aprūpinti visus savo malūnus vandeniu, bet jis nenori prarasti nei kiek vandens. Taškuose A–I yra užtvankos, neleidžiančios vandeniui tekėti žemyn (rodyklių kryptimi). Majus gali valdyti kiekvieną užtvanką atskirai vieną nuo kitos. Paspaudus užtvanką, ji bus uždaroma arba vėl atidaroma.



Spustelėkite užtvankas, kurias reikia uždaryti, kad vanduo tekėtų tik į malūnus.

Paaiškinimas

Turi būti uždarytos F, D ir H užtvankos. Tai vienintelis būdas aprūpinti tris malūnus vandeniu, jo nešvaistant. Sprendimą galima paaiškinti taip:

Užtvankos A, G ir I turi būti atidarytos, kitaip vanduo nepratekės į 1-ą malūną.

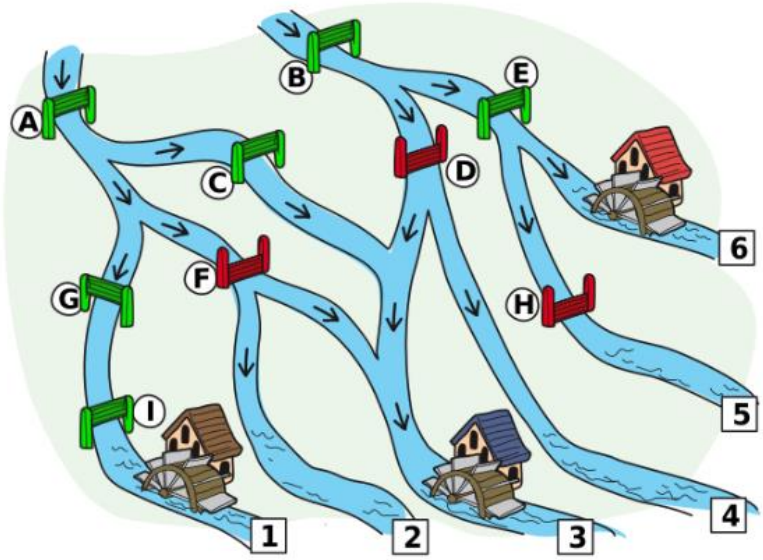
Taip pat reikia atidaryti B ir E užtvankas, priešingu atveju 6-as malūnas negaus vandens.

Kadangi A yra atidaryta, F užtvanka turi būti uždaryta, nes priešingu atveju vanduo ištekės 2-qja upe, kurioje nėra malūno.

Analogiškai, kadangi B atidaryta, D turi būti uždaryta, kitaip vanduo išbėgs 4-aja upe.

C užtvanka turi būti atidaryta, priešingu atveju 3-as malūnas negaus vandens.

Galiausiai, H turi būti uždaryta (nes B ir E yra atidarytos), kitaip 5-qja upe vanduo išbėgs.



Tai informatika!

Vandens srautą reguliuoja sąlygos. Pavyzdžiui, vanduo patenka į 2-ą upę, kai abi užtvankos A ir F atidarytos. Vanduo teka į 3-ą malūną, kai įvykdoma viena iš dviejų (arba abi) šių sąlygų: (1) A atidaryta ir viena iš dviejų užtvankų C arba F yra atidaryta arba (2) abi užtvankos, B ir D, yra atidarytos.

Šios sąlygos aprašomos naudojant logines operacijas IR arba ARBA. Sudėtinių reiškinių „X IR Y“ ir „X ARBA Y“ loginės reikšmės parodytos lentelėje:

X	Y	X IR Y	X ARBA Y
teisinga	teisinga	teisinga	teisinga
teisinga	klaidinga	klaidinga	teisinga
klaidinga	teisinga	klaidinga	teisinga
klaidinga	klaidinga	klaidinga	klaidinga

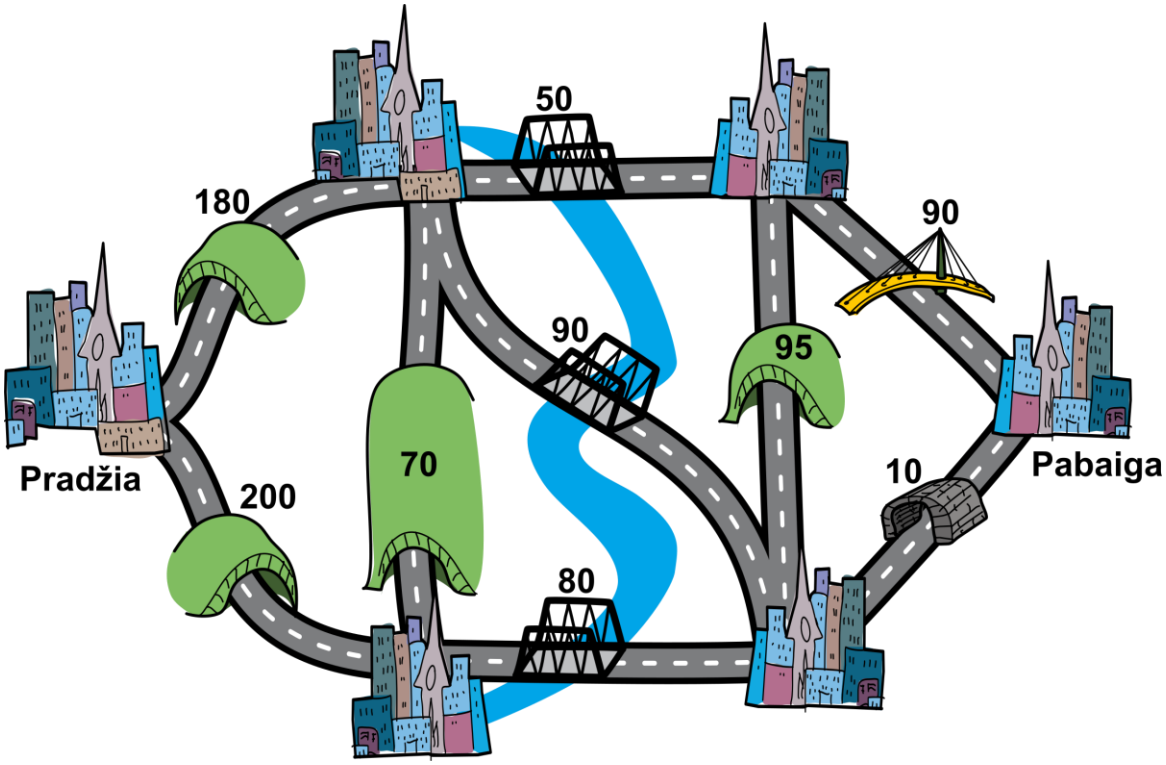
Informatikoje (taip pat ir matematikoje) teiginys „X ARBA Y“ taip pat laikomas teisingu, jei X ir Y yra teisingi.

Teiginys „vanduo išbėga 2-qja upe“ prilygsta „A atidaryta“ IR „F atidaryta“. Teiginys „vanduo išbėga 3-qja upe“ yra lygus („A atidaryta“ IR („C atidaryta“ ARBA „F atidaryta“)) ARBA („B atidaryta“ ir „D atidaryta“)).

Programuojant svarbu teisingai suformuluoti sąlygas. Loginės operacijos naudojamos sudėtinėms sąlygoms formuoti. Šios sąlygos naudojamos pasirinkimo komandoje „if” ir cikle „while”.

18. Geriausias maršrutas

Sunkvežimiai važinėja keliais, kurie parodyti žemėlapyje:



Prie tiltų ir tunelių nurodyti didžiausi leistini aukščiai sunkvežimių, kurie gali jais važiuoti.

Kokio didžiausio aukščio sunkvežimis gali nuvažiuoti keliais nuo pradžios iki pabaigos?

A. 50

B. 80

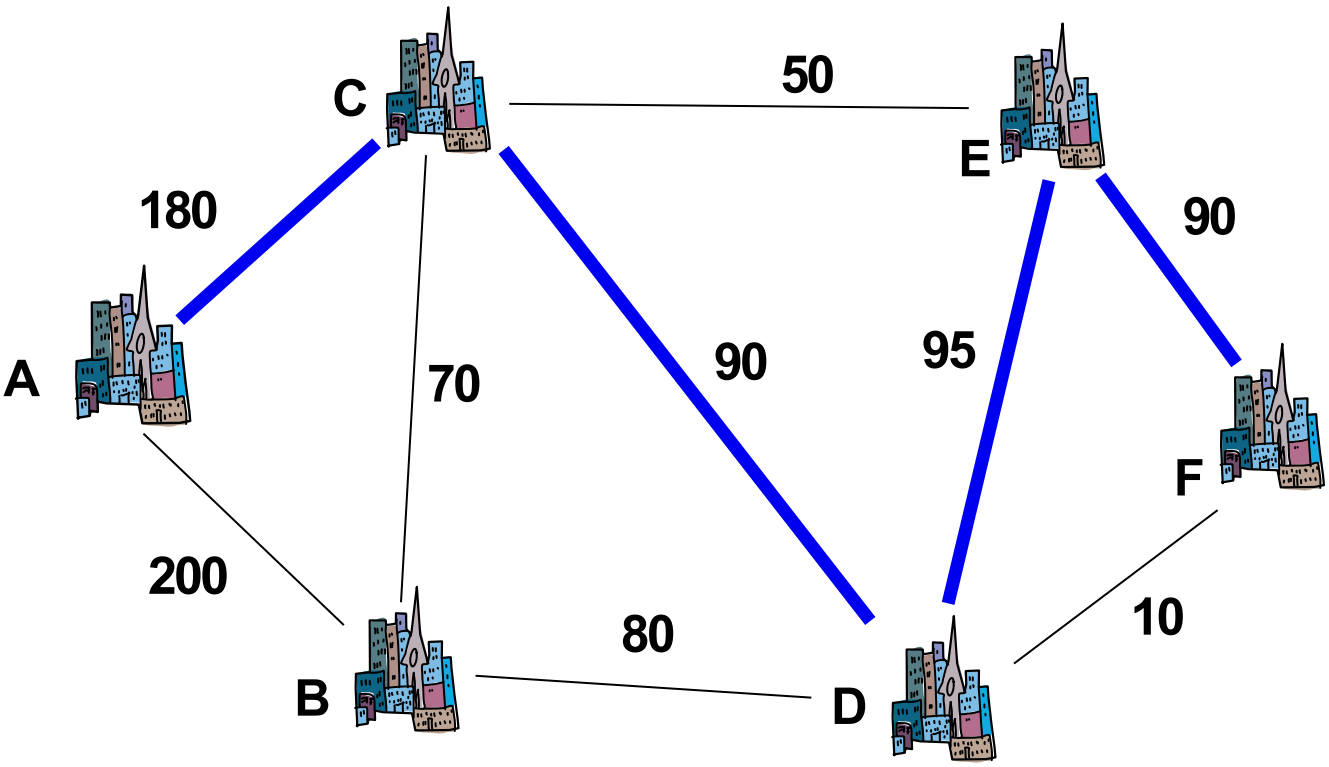
C. 90

D. 95

Paaiškinimas

Teisingas atsakymas: C.

Sunkvežimis, kurio aukštis yra 90, gali nuvažiuoti nuo pradžios (A) iki pabaigos (F) maršrutu A–C–D–E–F. Sunkvežimio, važiuojančio šiuo maršrutu, kelyje bus tokie aukščio ribojimai: 180, 90, 95 ir 90. Kadangi mažiausias skaičius visame maršrute yra 90, tai sunkvežimis, kurio aukštis 90, gali nuvažiuoti nuo pradžios iki pabaigos.



Alternatyvus (ilgesnis) būdas išspręsti šią užduotį – tikrinti visus įmanomus variantus nuo pradžios iki pabaigos.

Tai informatika!

Šios užduoties pagrindas – duomenų kiekio ribojimas perduodant duomenis kompiuteriniais tinklais (belaidžiu ryšiu ar kabeliais). Maršrutizatoriai ir algoritmai valdo duomenų srautą. Paprastai sakome, kad duomenų paketai perduodami iš vieno segmento į kitą. Svarbus rodiklis yra duomenų pralaidumo dydis, kuris nusako kaip greitai duomenų paketai gali būti perduodami tarp segmentų per tam tikrą laiką. Tai tas pats, kaip aukščio ribojimai sunkvežimiams, važiuojantiems per tiltus ir tunelius.

19. Kamuolio judėjimas

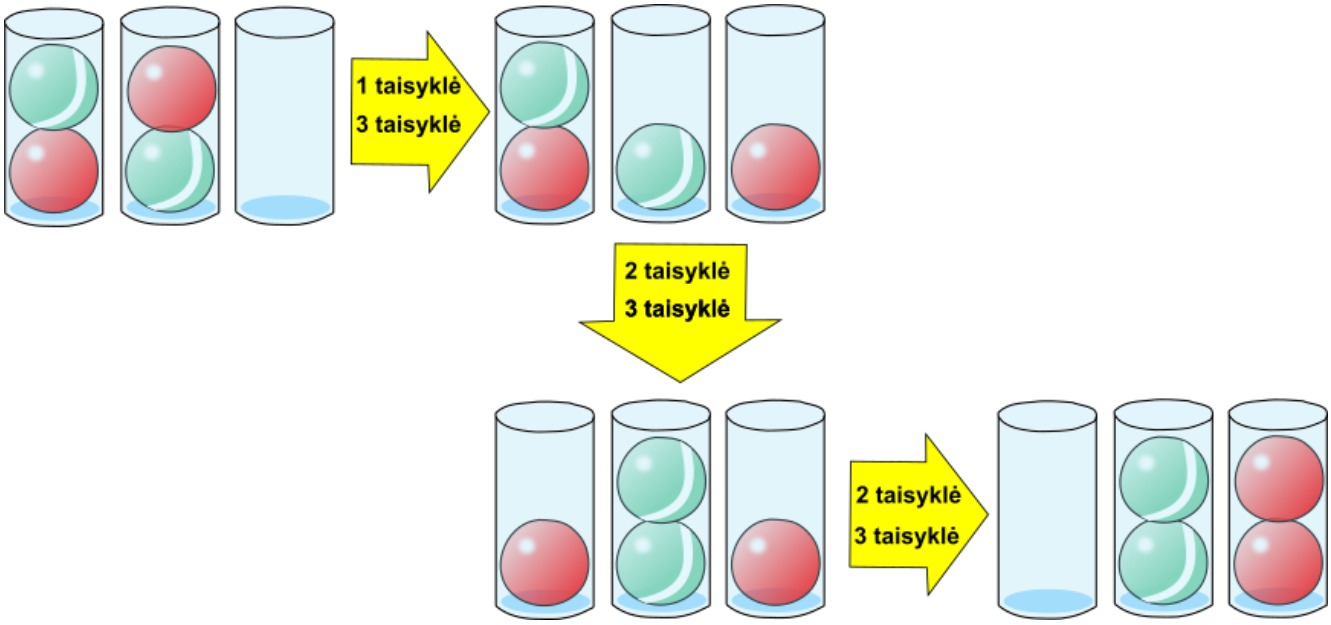
Bebrai žaidžia žaidimą su cilindruose judančiais kamuoliais. Šio žaidimo tikslas yra suskirstyti tos pačios spalvos kamuolius į tą patį cilindrą. Žaidimo taisyklės tokios:

1 taisyklė: Kamuolys gali būti perkeltas į kitą cilindrą, jei jame nėra kamuolių.

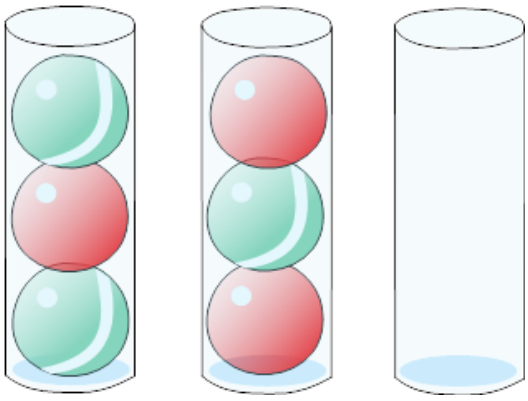
2 taisyklė: Jei cilindre yra vietos, ant jame esančio kamuolio iš kito cilindro galima perkelti tik tos pačios spalvos kamuolį.

3 taisyklė: Vienu žingsniu galima perkelti tik vieną kamuolį iš cilindro viršaus.

Pavyzdys:



Duota tokia situacija:

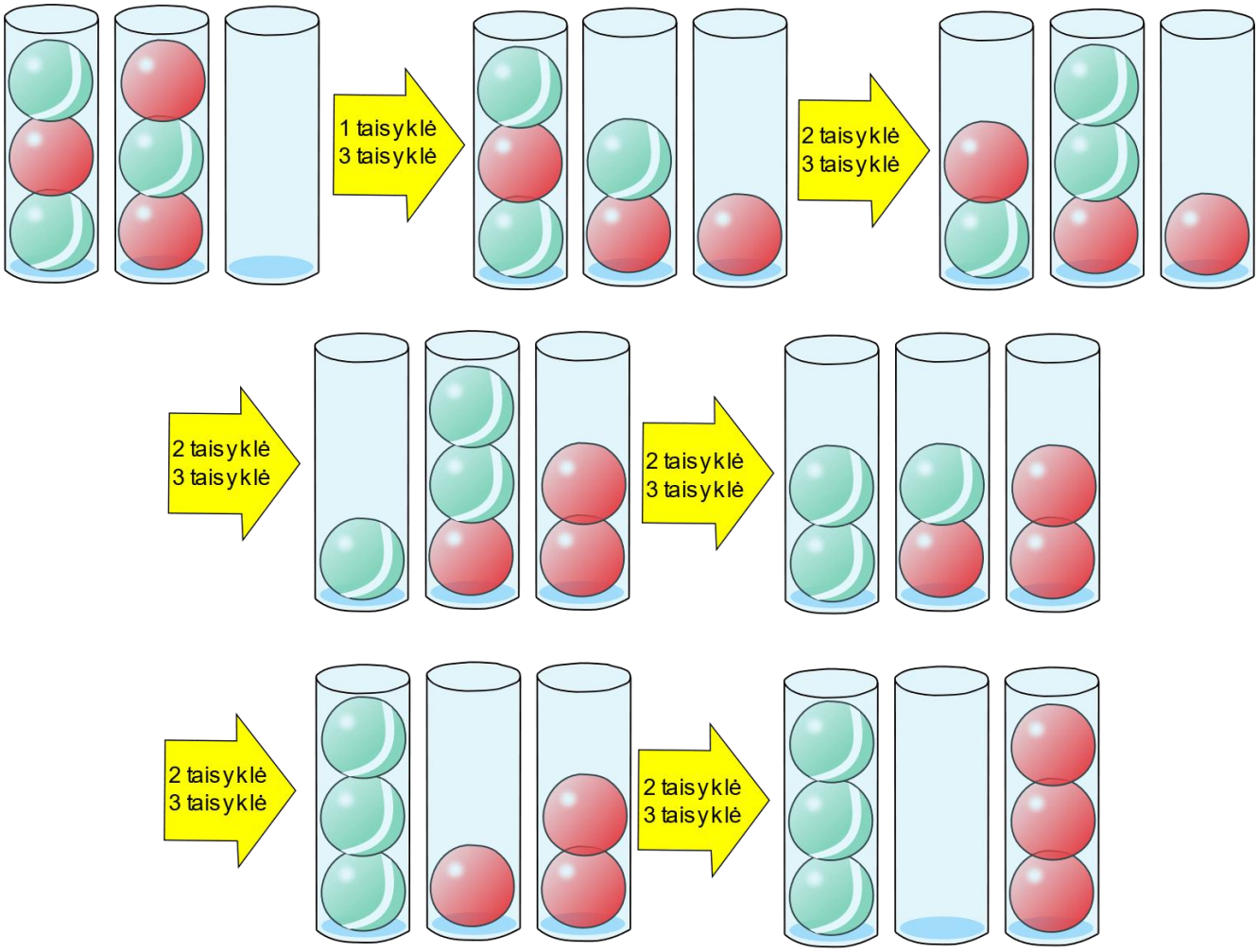


Kiek žingsnių mažiausiai reikia atlikti, jei norite laimėti žaidimą?

Paaiškinimas

Teisingas atsakymas: 6.

Kamuolių perkėlimo tvarka pateikiama paveiksle.



Tai informatika!

Informatikoje duomenų saugojimui gali būti naudojami vadinamieji dėklai. Dėklų ypatybė yra ta, kad galite pasiekti ir pašalinti tik vėliausiai pridėtus duomenis.

Pavyzdyje esantys cilindrai veikia kaip dėklai: kamuolį galima pridėti tik cilindro viršuje, pašalinti kamuolį irgi galima tik iš viršaus. Prie kitų kamuolių negalima patekti.

20. Ilgiausia seka

Duota 16 simbolių ilgio seka, sudaryta iš dviejų skirtingų rūšių simbolių.

Bet kuriuos du simbolius sekoje galima pakeisti kitos rūšies simboliu.

Kokia yra ilgiausia įmanoma vienodų simbolių seka po pakeitimo?

A) 7

B) 8

C) 9

D) 10

Paaiškinimas

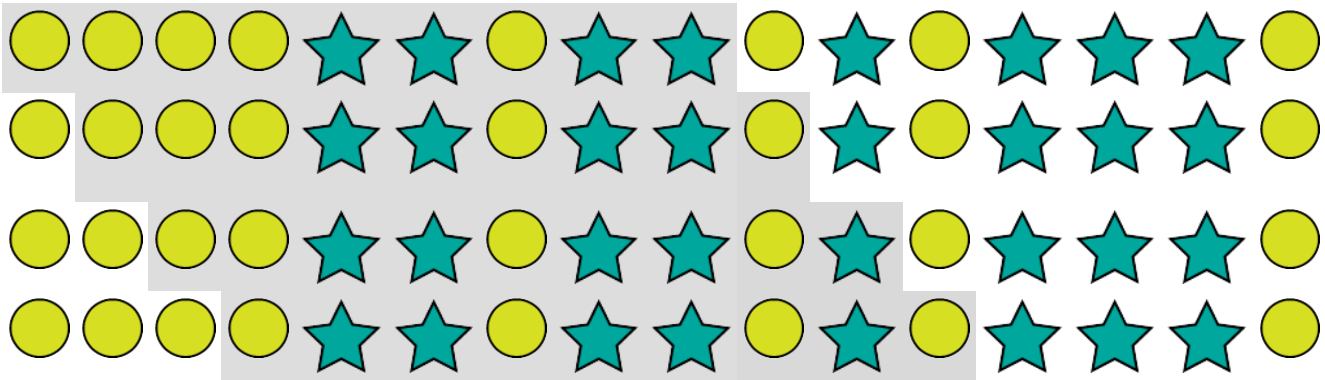
Teisingas atsakymas: 8.

Norėdami tai pagrįsti, turime parodyti, jog (1) 8 simbolių ilgio seka yra įmanoma ir (2) ilgesnė nei 8 simbolių ilgio seka neįmanoma.

Įrodyti, kad 8 simbolių seka yra įmanoma galime taip:

Norėdami įrodyti, jog ilgesnė nei 8 simbolių seka neįmanoma, paimkime bet kokią 9 simbolių ilgio seką. Kadangi mes galima pakeisti tik 2 simbolius, tai toje 9 simbolių ilgio sekoje 7 simboliai jau turi būti vienodi.

Yra įmanomos aštuonios 9 simbolių ilgio sekos, kelios iš jų yra parodytos. Kadangi nei vienoje 9 simbolių ilgio sekoje nėra 7 vienodų simbolių, tai 9 vienodų simbolių seka yra neįmanoma.



Kadangi 9 vienodų simbolių seka yra neįmanoma, tai ir ilgesnė nei 9 taip pat yra neįmanoma.

Tad ilgiausia įmanoma vienodų simbolių seka yra 8 simboliai.

Tai informatika!

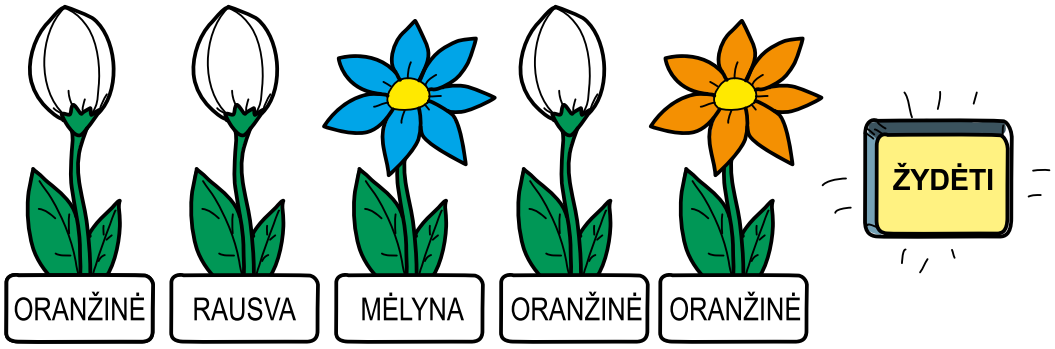
Ši užduotis – ilgiausio poaibio ieškojimas pagal tam tikrus kriterijus.

Informatikoje poaibio ieškojimas yra dažnai naudojamas, ypač kai reikia surasti bendrą dviejų aibių poaibį.

Bendro poaibio radimas padeda aptikti plagijavimą, taip pat suglaudinti duomenis panaikinant pasikartojančią informaciją.

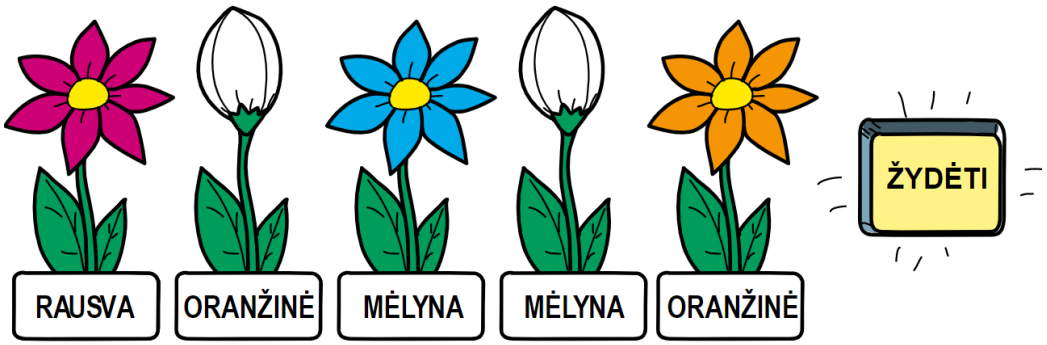
21. Gėlių spalvos

Nojus žaidžia kompiuterinį žaidimą. Kompiuteris slapčiomis parinko spalvas penkioms nepražydusioms gėlėms. Galimos trys spalvos: mėlyna, oranžinė, rausva. Gėlių spalva nekinta.



Nojus spėja gėlės spalvą parinkdamas pavadinimą po gėle. Paspaudžia mygtuką „Žydėti“ – pražysta tik tos gėlės, kurių spalvas Nojus atspėjo.

Nojus toliau spėja nepražydusių gėlių spalvas ir vėl spaudžia mygtuką „žydėti“:



Kokias spalvas kompiuteris parinko visoms gėlėms?

A. 

B. 

C. 

D . 

Paaiškinimas

Teisingas atsakymas: C. 

Spėjamos visų gėlių spalvos. Po dviejų spėjimų pražydo trys gėlės. Taigi žinome kompiuterio parinktas spalvas pirmai, trečiai ir penktai gėlėms. Pirmosios gėlės spalva yra rausva, vadinasi, A atsakymas klaidingas.

Kai Nojus bandė atspėti antrosios gėlės spalvą, pirmuoju atveju jis parinko rausvą spalvą, o antruoju – oranžinę. Abiem atvejais gėlės pumpuras neišsiskleidė. Kadangi iš viso yra trys spalvos, tai antrojo žiedo spalva turi būti mėlyna. Tuomet D atsakymas yra klaidingas.

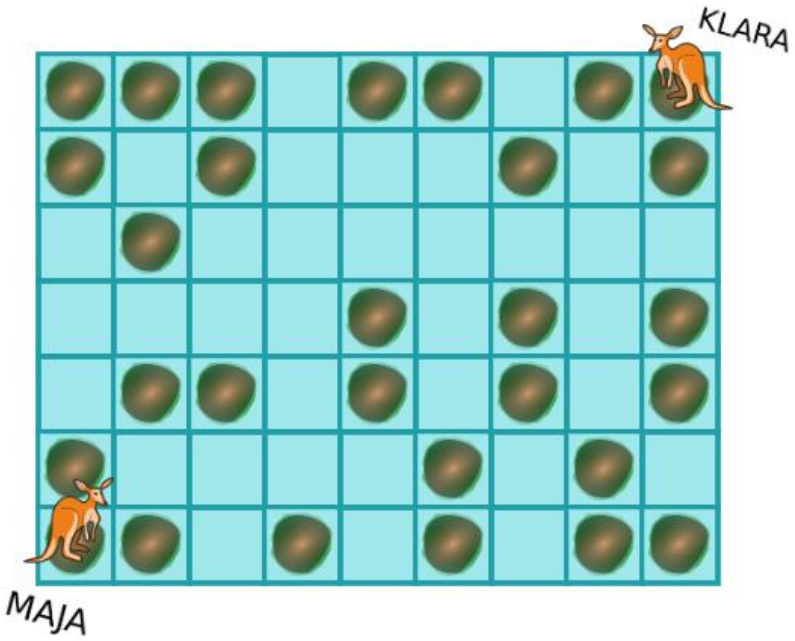
Analogiškai Nojus parinko oranžinę ir mėlyną spalvas ketvirtajai gėlei, kuri irgi nepražydo. Vadinasi, šios gėlės spalva turi būti rausva. Taigi ir B atsakymas yra neteisingas. Lieka teisingas C atsakymas.

Tai informatika!

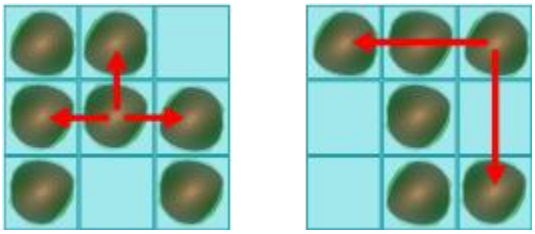
Uždavinių sprendimui svarbu gebėti prognozuoti įvykių baigtį. Ši užduotis yra suprastintas stalo žaidimo „Mastermind“ (klasikinio kodo laužymas) variantas. Po kiekvieno spėjimo žaidėjas gauna visą informaciją apie gėles. Jeigu kiekvieną kartą spėjant gėlių spalvos parenkamos skirtingos, tuomet trečiu spėjimu galima atspėti teisingą spalvą.

22. Kengūra

Kengūra Maja nori nušuoliuoti pas kengūrą Klarą. Kelias tarp šių kengūrų eina salelėmis. Pelkę ir saleles galima pavaizduoti, kaip parodyta paveiksle.



Maja gali atlikti tik dvejopus šuolius: trumpą ir ilgą. Trumpu šuoliu kengūra iš vieno langelio gali nušokti į bet kurį iš keturių besiribojančių langelių. Ilgu šuoliu Maja gali šokti tiesiai ir peršokti bet kurį iš besiribojančių keturių langelių. Žemiau kairiajame paveikslėlyje pavaizduoti trumpi šuoliai, dešiniajame – ilgi.

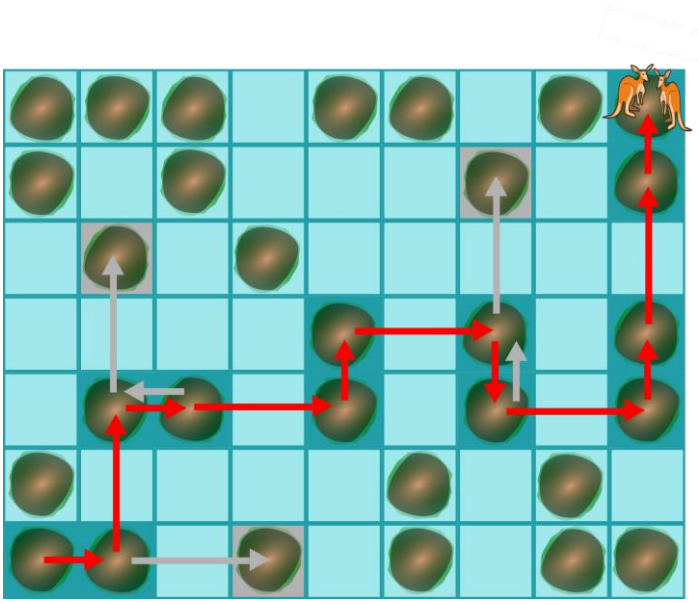


Maja negali atlikti jokių kitų šuolių: šokti įstrižai, peršokti dviejų ar daugiau langelių ir pan. Ilgi šuoliai vargina labiau ir yra pavojingesni už trumpus, todėl Maja negali atlikti dviejų ilgų šuolių iš eilės.

Raskite kelią, kuriuo Maja galėtų pasiekti Klarą.

Paaiškinimas

Teisingas atsakymas pavaizduotas paveiksle.



Galimi ir kitokie sprendimai, kai Majai tektų į kai kurias saleles šokti po keletą kartų.

Yra tik trys salelės (pažymėtos pilka rodykle), kurias Maja gali pasiekti, tačiau jos nepatenka į Majai reikalingą kelią – iš šių salelių bandant eiti toliau, Majai tektų atlikti po du ilgus šuolius iš eilės. Atmetus šias tris saleles, iš likusių salelių, kurias Maja gali pasiekti, suformuojamas reikiamas kelias.

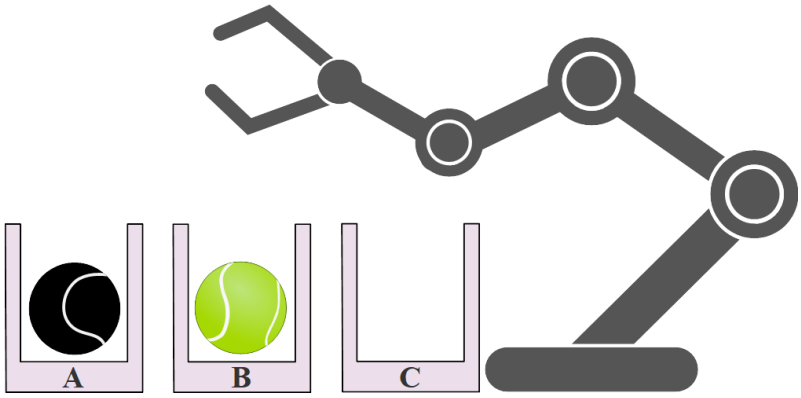
Tai informatika!

Užduotyje naudojama paieška į plotį – modifikuotas Lee algoritmas. Lee algoritmas imituoja bangos judėjimą užliejant gretimus langelius. Šiame uždavinyje langeliai laikomi gretimais ir tuomet, kai kengūra gali juos pasiekti ir dviem šuoliais: pirma ilgu, po to – trumpu.

Užduoties sąlyga draudžia du ilgus šuolius iš eilės, todėl uždavinio negalima išspręsti naudojant dinaminį programavimą.

23. Roboto ranka

Yra du kamuoliai, kurių vienas yra A dėžutėje, o kitas – B dėžutėje. C dėžutė tuščia.



Roboto ranka iš eilės atlieka šiuos veiksmus:

1. Paima kamuolį iš A dėžutės ir padeda į C dėžutę.
2. Paima kamuolį iš B dėžutės ir padeda į A dėžutę.
3. Paima kamuolį iš C dėžutės ir padeda į B dėžutę.

Kurie iš šių teiginių bus teisingi, roboto rankai baigus atlikinėti veiksmus?
Galima pasirinkti daugiau nei vieną atsakymą.

A) Kamuoliai apsikeitė vietomis.

B) A dėžutėje yra du kamuoliai.

C) B dėžutėje yra du kamuoliai.

D) A dėžutė yra tuščia.

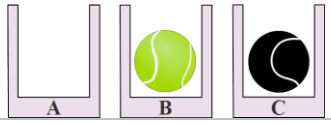
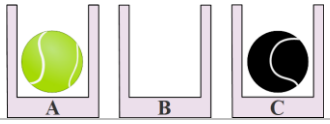
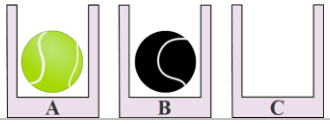
E) C dėžutė yra tuščia.

F) Niekas nepasikeitė. Abu kamuoliai grįžo į savo vietas.

Paaiškinimas

Atsakymai A ir E yra teisingi.

Peržiūrėkite kamuolių padėtis po kiekvieno įvykdyto veiksmo:

Po 1-o veiksmo	*Po 2-o veiksmo*	*Po 3-o (paskutinio) veiksmo*
		

Tai informatika!

Šioje užduotyje pateikta informatinio mąstymo koncepcija yra algoritmai. Norėdami išspręsti šią užduotį, turite atlikti veiksmus nurodyta eilės tvarka, sekti kamuolių padėtį po kiekvieno žingsnio.

Roboto ranka keičia kamuolių padėtį. Kadangi vienu metu galima paimti tik vieną kamuolį, roboto ranka turi padėti pirmąjį kamuolį, kad galėtų paimti kitą. Todėl mainų procesui reikia trijų vietų.

Norint sukeisti dviejų kintamųjų *a* ir *b* reikšmes vietomis, pirmojo kintamojo reikšmė laikinai suteikiama trečiam kintamajam *c*:

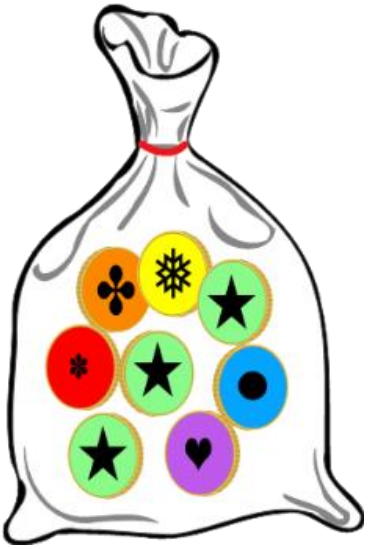
```
c = a
a = b
b = a
```

Skaitinių sveikųjų skaičių reikšmėms taikoma procedūra be naujo trečiojo kintamojo:

```
a = a + b
b = a - b
a = a - b
```

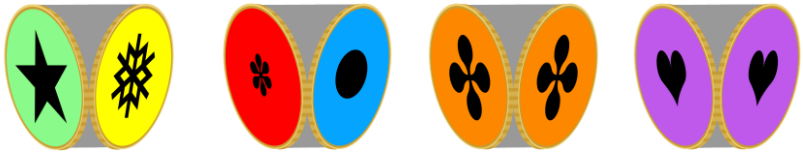
Kai kurios programavimo kalbos, pvz., „Python“, leidžia kelias kintamojo reikšmes (tam naudojamas specialus duomenų tipas *tuples*): a, b = b, a.

24. Monetų maišas



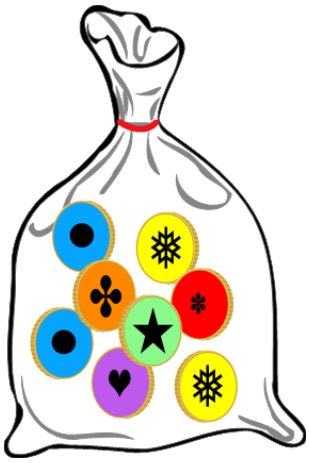
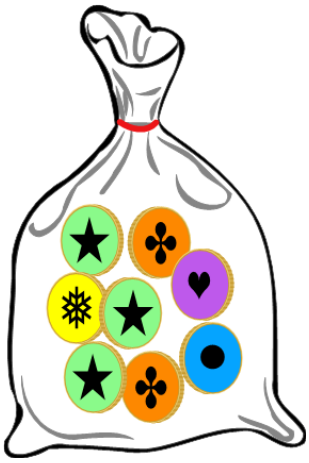
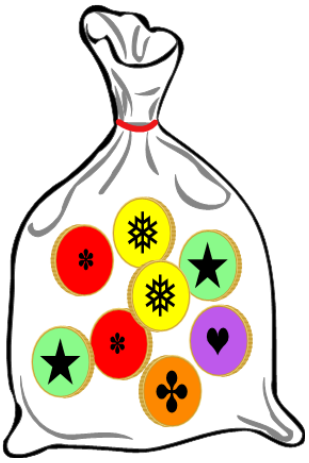
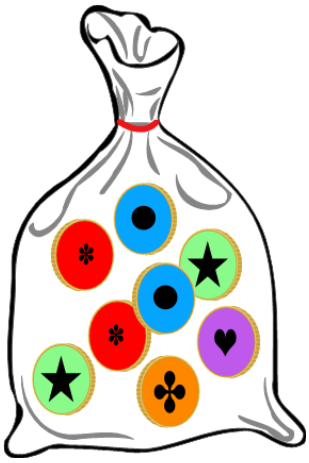
Sauliaus dvare vartojamos tik 4 skirtingų tipų monetos.

Galite matyti abi šių monetų puses ir Sauliaus maišą su monetomis.



Vėliau monetų maišas buvo supurtytas.

Kuris iš šių maišų priklauso Sauliui?

A.	B.	C.	D.
			

Paaiškinimas

Teisingas atsakymas: C.

Sauliaus maiše yra monetos, kurių kiekiai nurodyti lentelėje.

				
Sauliaus maišas	4	2	1	1
A maišas	3	3	1	1
B maišas	4	1	2	1
C maišas	4	2	1	1
D maišas	2	4	1	1

Tik C maiše yra tiek pat skirtingų rūšių monetų, kiek ir Sauliaus maiše. Vadinasi, tai yra teisingas sprendimas.

Tai informatika!

Atlikdami šią užduotį turite atpažinti monetų tipus nematydami abiejų monetų pusių. Turite tik neišsamią informaciją. Realaus pasaulio objektai su pagrindinėmis savybėmis yra aprašomi kompiuterių sistemose. Dažnai pakanka žinoti tik dalį šių savybių, kad būtų galima atpažinti objektą. Fotoaparatas autonominėje transporto priemonėje mato tik realybės dalis, tačiau kompiuterinė sistema turi sugebėti atpažinti kitas transporto priemones ir eismo dalyvius bei tinkamai reaguoti į atitinkamą eismo situaciją. Dirbtinis intelektas kompiuterinėse sistemose, kaip ir žmonės, pamažu išmoksta vis tiksliau atpažinti objektus iš fragmentų.

Kai kurie pasaulio informacijos elementai (istorijos, pokalbiai, žinutės, pirkinių sąrašai) gali būti skirtingo ilgio ir skirtingo sudėtingumo (vadinamoji nestruktūrizuota informacija). Kompiuterių mokslininkai, tvarkydami informaciją, dažnai turi sukonstruoti daiktų struktūrą. Kartais į tam tikras savybes nereikia kreipti dėmesio, o skirtingai atrodantys dalykai turi būti laikomi lygiaverčiais. Tai abstrakcijos pavyzdys.

Atliekant šią užduotį monetų maišas (arba kelių rinkinių rinkinys) naudojamas kaip nestruktūrizuotų duomenų pavyzdys (nėra konkrečių nuorodų į monetas ir gali būti kelios to paties tipo monetos). Tačiau yra ir sudėtingesnių realaus pasaulio pavyzdžių. Prasmės išgavimas iš žmogaus kalbos yra viena labai sunki, bet svarbi informatikos užduotis, susijusi su nestruktūrizuotais duomenimis.

Pavyzdžiui, įsivaizduokite, jog žmonių klausiama „Kas jums labiausiai patiko šiame filme?“ ir įvairūs žmonės atsakė šiomis frazėmis:

„Man patiko rezultatas.“

„Šio filmo muzika.“

„Nauja garso patirtis.“

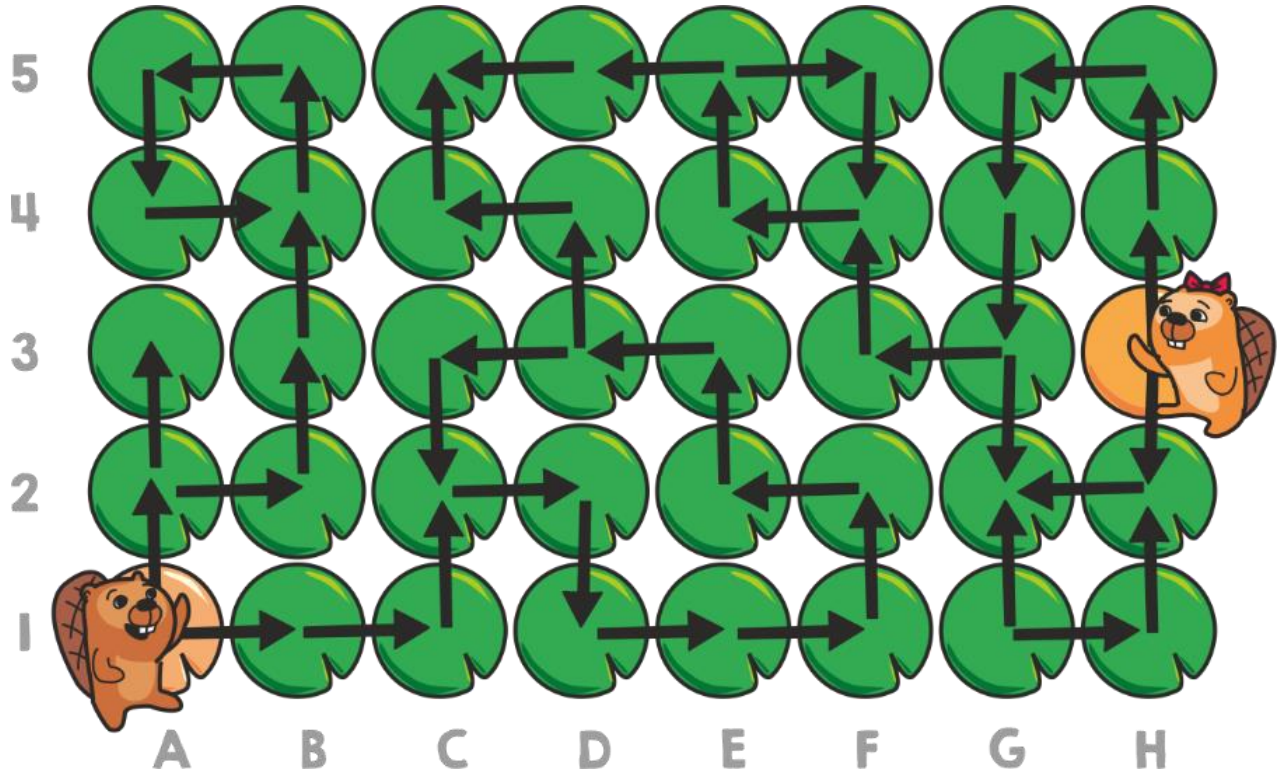
„Aš atpažinau savo mėgstamiausią dainą.“

Kompiuterinė programa, analizuojanti šių žmonių atsakymus, turėtų pripažinti, kad šiame kontekste visi šie teiginiai turėtų būti pateikiami kaip lygiaverčiai, nors juose vartojami skirtingi žodžiai.

25. Ar jie susitiks?

Du bebrai – Balys ir Nora – ežere vaikšto lelijų lapais. Jie gali pereiti tik ant to lapo, į kurį rodo rodyklė.

Spustelėk lapą, ant kurio bebrai susitiks.



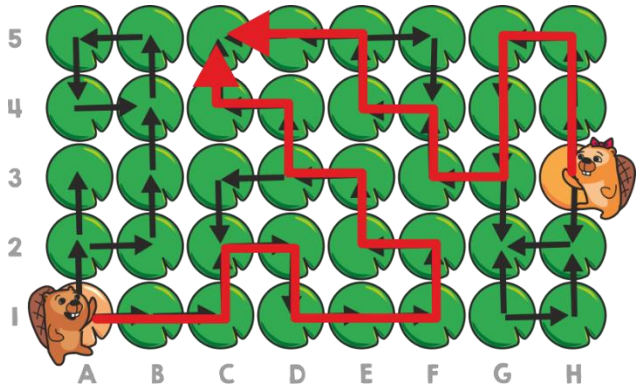
Paaiškinimas

Teisingas atsakymas: bebrai gali susitikti ant C5 lapo.

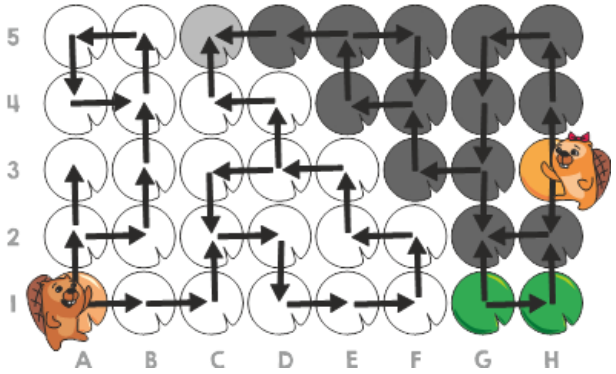
Balys turi du pasirinkimus: iš savo pradinės pozicijos eiti į viršų arba dešinėn. Jei Balys eis į viršų, tai jis galiausiai prieis akligatvį A3 arba turės vaikščioti ratu B4 → B5 → A5 → A4 ir t. t. Jei Balys pradės eidamas į dešinę (B1), jis eis toliau iki D3. Ties D3 jis gali eiti į kairę, kur galiausiai jis pradės vaikščioti ratu, arba eiti į viršų, į lapą C5.

Nora taip pat turi du pasirinkimus: eiti žemyn, kas galiausiai ją atves į kelio pabaigą G2, arba judėti į viršų, kol pasieks G3. Nuo G3 ji gali eiti atgal į G2, kur baigiasi kelias, arba eiti į kairę, galiausiai prieis išsišakojimą ant E5 lapo. Nuo šio lapo Nora gali keliauti į ratą, kuris galiausiai ją atves atgal į E5, arba eiti į kelio pabaigą C5.

Jau žinome, kad Balys taip pat gali nukeliauti iki C5. Todėl matome, jog C5 yra galimas susitikimo taškas. Paveiksle parodyta, kaip abu bebrai gali pasiekti C5:



Tai dar neįrodo, jog bebrai negali susitikti ant F4 ir C2 lapų. Kitame paveiksle parodyti visi įmanomi bebrų keliai, tad galime matyti, jog vienintelis lapas, ant kurio abu bebrai gali užlipti, yra C5.



Tai informatika!

Įdėmiau pažiūrėję į pastarąjį paveikslą matome, jog jei bebrai prieidavo kelio galą arba pradėdavo eiti ratu, jie grįždavo iki paskutinio išsišakojimo ir eidavo kitu keliu. Tokiu būdu mes apėmėme visus įmanomus kelius. Šį metodą naudoja kompiuterių mokslininkai, spręsdami sudėtingas problemas. Ši procedūra bando gauti sprendimą pažingsniui, dažniausiai tokio tipo uždavinių išsišakojimuose būna keli pasirinkimai, procedūra pasirinks vieną kelią, bet tuo pačiu prisimins ir kitus variantus, tad priėjus kelio galą ji galės grįžti iki pastarojo pasirinkimo ir bandyti kitą kelią. Šis metodas vadinamas grįžimo metodu.

26. Labirintas

Ragana Luna tyrinėja labirintą, kuriame slypi penki lobiai: monetos, rubinas, burtų knyga, skrynelė ir stebuklingas gėrimas.

Luna nežino, kurį lobį iš penkių rinktis, tad nusprendžia laikytis tokių taisyklių:

- Visada judėti žemyn (↓) – šiam veiksmui taikoma pirmenybė;
- Jei nėra kelio žemyn, judėti į dešinę (→);
- Jei nėra kelio žemyn ar į dešinę, judėti į kairę (←);
- Nesisukti atgal ir neskristi į viršų (↑), kol nebus pasiektas lobis.

Kurį lobį ras Luna?

A. 1

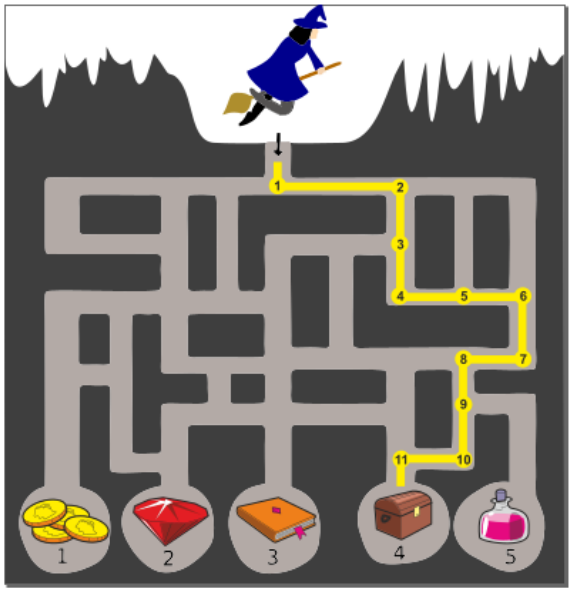
B. 2

C. 3

D. 4

E. 5

Paaiškinimas



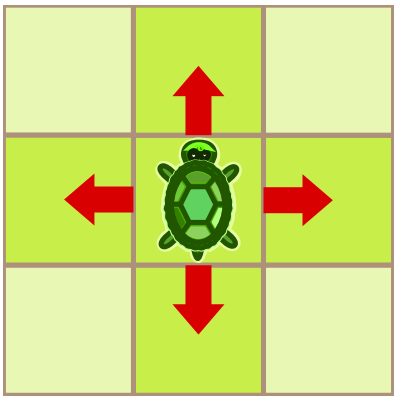
Teisingas atsakymas: D. Kelias pavaizduotas paveiksle.

Kai Luna įskrenda į urvą, ji turi judėti žemyn pagal pirmą taisyklę. Pirmame sutiktame išsišakojime ji gali pasirinkti judėti į kairę ar į dešinę. Pagal 2-ą taisyklę ji turi skristi į dešinę. Antrame išsišakojime ji turi skristi žemyn, kadangi šiam veiksmui visada taikoma pirmenybė. 3–6-ame išsišakojimuose taikomos pirmos dvi taisyklės. 7-ame išsišakojime nėra kelio žemyn ar į dešinę. Todėl Luna turi judėti į kairę. Taikydama visas keturias taisykles, ji pagaliau pasiekia 4-ą lobį – skrynelę.

Tai informatika!

Sąlyginiai sakiniai informatikoje yra esminiai ir taikomi vykdant įvairias programas, laikantis tam tikrų sąlygų ar taisyklių. JEI sąlyga teisinga, tai programa (veiksmas) vykdomas, KITAIP – vykdoma kita programa (veiksmai). Šiame uždavinyje turime tris taisykles. JEI pasiekiamas labirinto išsišakojimas ir yra kelias žemyn, judėti žemyn. Priešingu atveju, JEI yra kelias į dešinę, judėti į dešinę. KITAIP – judėti į kairę. Lunai negalima skristi į viršų.

27. Vėžlio kelias



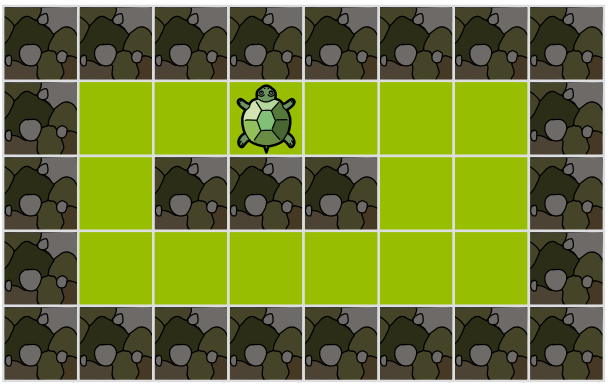
Vėžlys gyvena nedideliame sode. Sodas padalintas į kvadratinius laukelius, padengtus arba žole, arba akmenimis. Vėžlys gali judėti iš vieno žolės laukelio į kitą kaip parodyta paveikslėlyje. Ant akmenų vėžlys lipti negali.

Dėl savo lėto būdo vėžlys sudaro maitinimosi kelią, kuriuo pereina VISUS žolės laukelius, į kiekvieną užsukdamas TIK VIENĄ kartą.

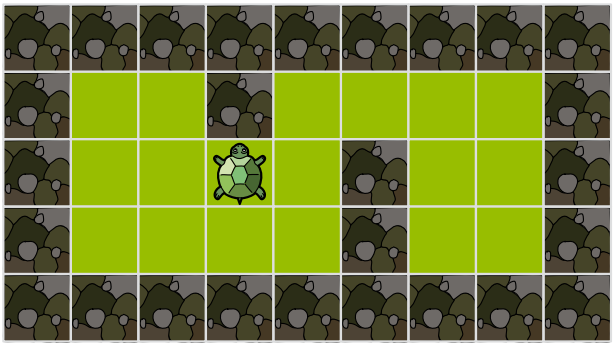
Kuriame sode toks kelias neįmanomas?

A.

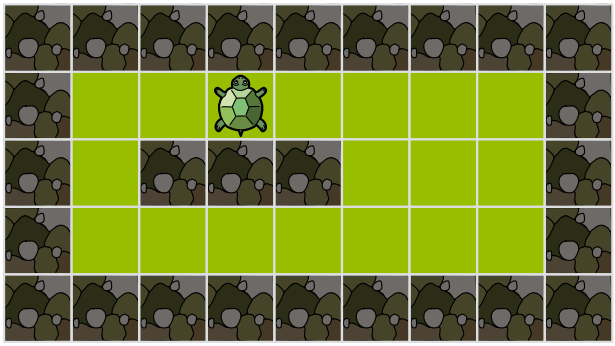
B.



C.

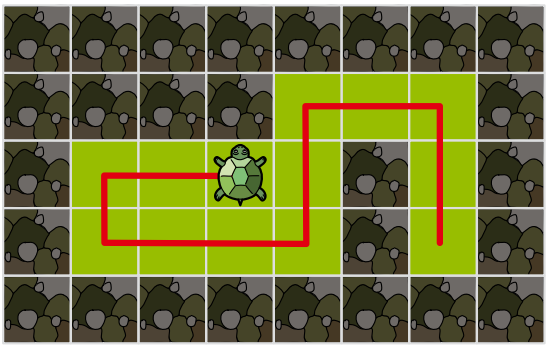


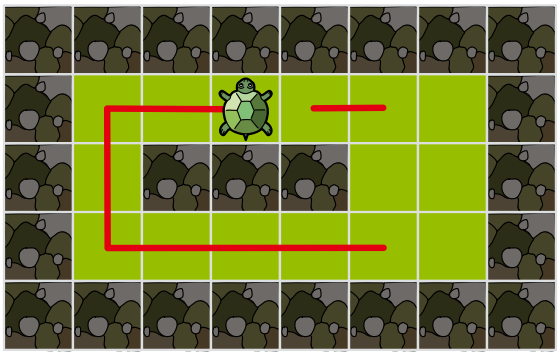
D.

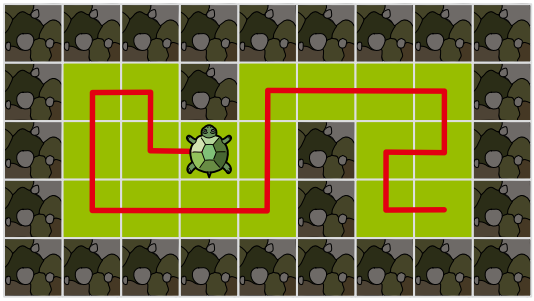


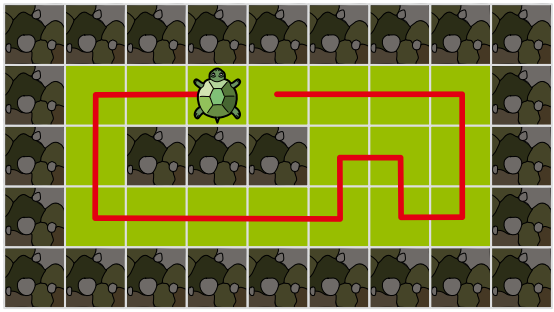
Paaiškinimas

Atsakymas:

A. 

B. 

C. 

D. 

Sode B nėra tokio kelio, kuriuo vėžlys galėtų aplankyti visus laukelius nesilankydamas nei viename iš jų du kartus. Tokiam keliui turėtų priklausyti dvi kelio dalys, tačiau vėžlys turėtų pereiti likusius 6 laukelius aplankydamas kiekvieną iš jų lygiai vieną kartą. Be to, šiuos langelius vėžlys turėtų pereiti pradėdamas viename jau pažymėto kelio gale ir pabaigdamas kitame. Deja, tokio kelio sudaryti tiesiog neįmanoma.

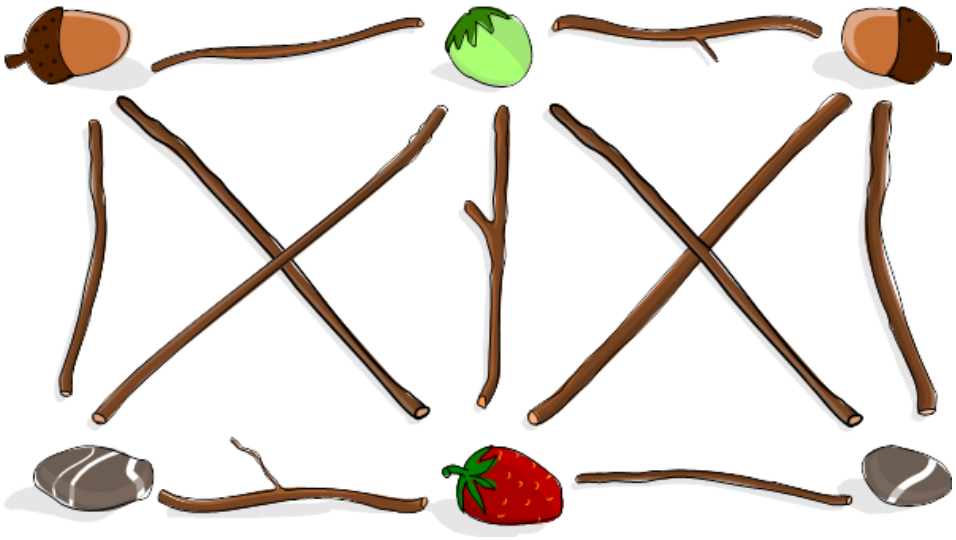
Tai informatika!

Vėžliai ieško maitinimosi kelio savo soduose kiekvieną laukelį aplankydami tik kartą. Tai yra gerai žinoma informatikose problema, tik vadinasi ji ne vėžlių, o Hamiltono maršruto uždaviniu. Į vėžlio sodą galima žiūrėti taip: žolės laukelius galime laikyti viršūnėmis (pažymėti taškais), o visus būdus pereiti iš vieno laukelio į kitą laikyti briaunomis (pažymėti linijomis tarp viršūnių). Tokius darinius matematikai ir informatikai vadina grafais.

XIX a. matematikas Viljamas Hamiltonas tyrinėjo kelius, einančius per visas viršūnes lygiai vieną kartą ir grįžtančius į pradinę viršūnę. Sudėtingesnėse situacijose rasti tokį Hamiltono ciklą yra itin sunku. Taip pat sunku rasti ir Hamiltono maršrutą, kuriame (kaip ir vėžlio sode) grįžti į tą pačią viršūnę nebūtina. Informatikoje vis dar nėra atrasta algoritmo, kuris bet kokiam grafui galėtų efektyviai (per pakankamai trumpą laiką) rasti Hamiltono ciklą ar Hamiltono maršrutą.

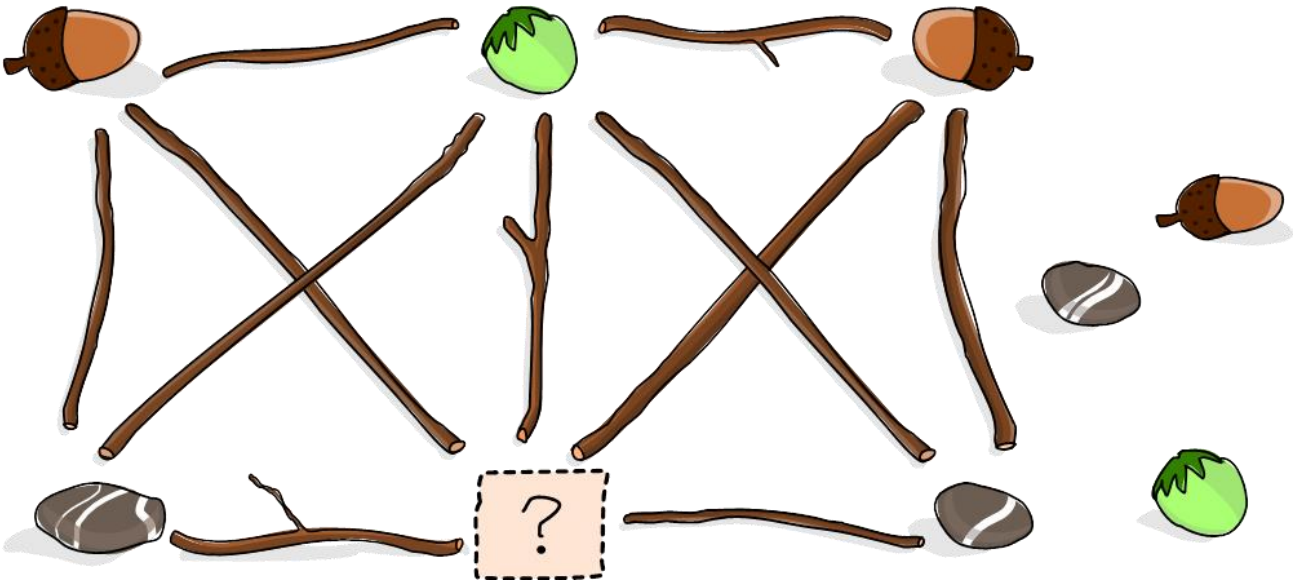
28. Braškių vagis

Onutė žaidžia lauke – kuria raštus iš gilių, riešutų, akmenukų ir braškių, tarp kurių ji deda lazdeles, laikydamasi vienos griežtos taisyklės: lazdelė gali būti dedama tik tarp skirtingų daiktų. Štai vienas iš Onutės raštų:

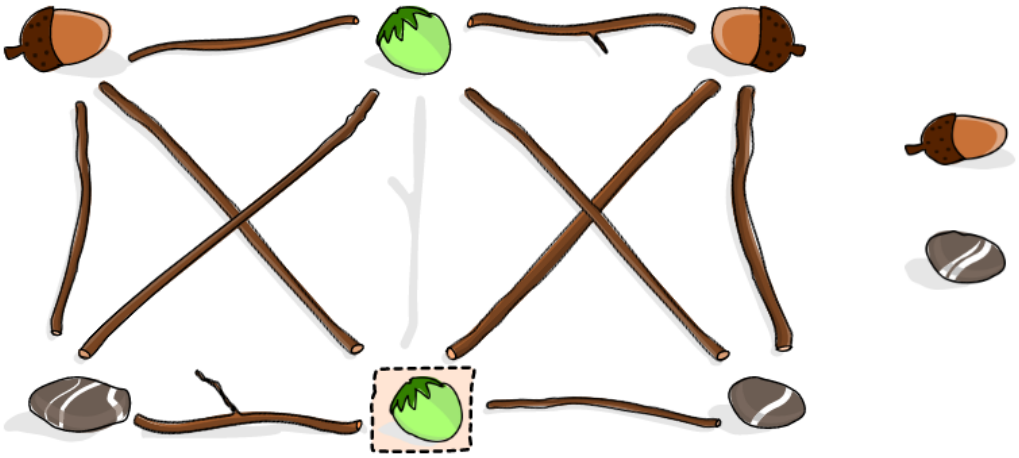


Onutės sesuo Zita, pamačiusi šį raštą, suvalgė braškę! Kad paslėptų pėdsakus, ji vietoj braškės padėjo kažką kitą (akmenuką, gilę ar riešutą). Ji taip pat nuėmė vieną lazdelę, kad griežtoji taisyklė nebūtų pažeista.

Kuo Zita pakeitė braškę ir kurią lazdelę nuėmė?



Paaiškinimas



Zita pakeitė braškę riešutu ir nuėmė lazdelę, jungiančią riešutus, nes ji pažeistų griežtąjį taisyklę (jungtų du tokius pat daiktus).

Keisti braškę bet kuriuo kitu daiktu negalima, nes taisyklei išlaikyti reikėtų nuimti daugiau nei vieną lazdelę.

Jei Zita pakeistų braškę gile, ji turėtų nuimti 2-ą ir 4-ą lazdeles.

Jei Zita pakeistų braškę akmenuku, ji turėtų nuimti 1-ą ir 5-ą lazdeles.

Tai informatika!

Onutės struktūra iš tiesų yra grafas, kurį sudaro viršūnės (daiktai) ir jas jungiančios briaunos (lazdelės). Dvi viršūnės, kurias jungia briauna, vadinamos kaimyninėmis.

Viršūnių poaibis, kuriame kiekviena viršūnė yra kaimyninė, vadinamas klika. Onutės raštas – tai dvi 4 dydžio klikos.

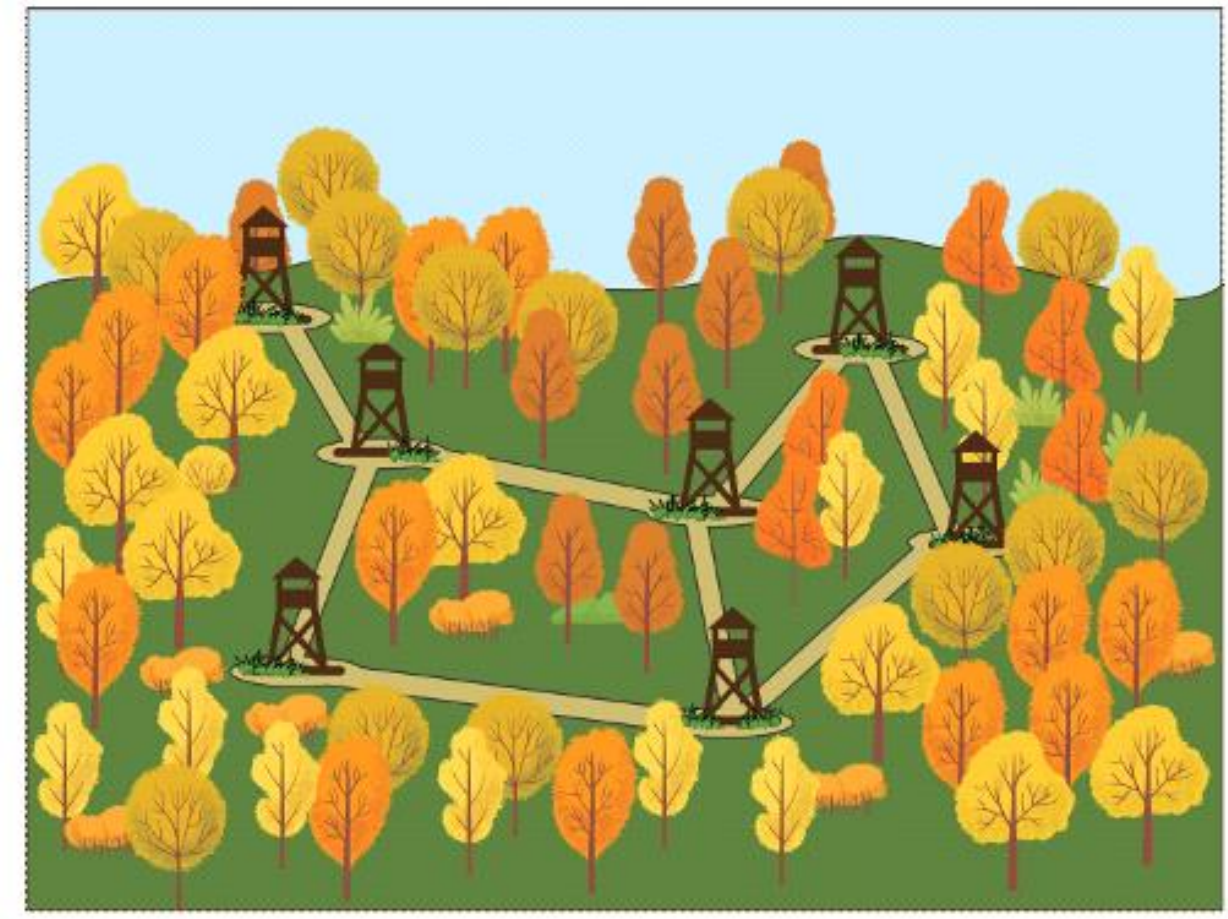
Braškės pašalinimas yra tarsi bandymas nuspalvinti Onutės grafą naudojant daugiausiai tris spalvas. Tai neįmanoma klikoms, kurių dydis yra 4, todėl Zita ir turi nuimti vieną lazdelę.

Grafo spalvinimo minimaliu spalvų skaičiumi problema turi daug taikymo sričių: sporto varžybų tvarkaraščių sudarymas, svečių susodinimas, sudoku galvosūkių sprendimas ir kt.

29. Miško stebėjimas

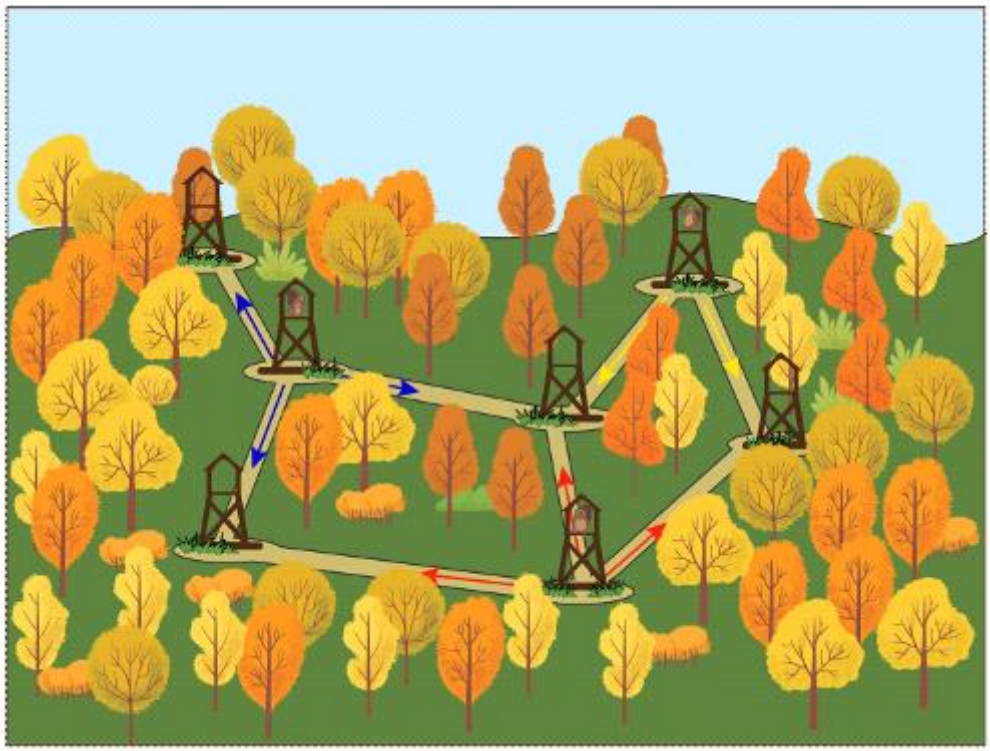
Girininkai stebi žvėris, kurie vaikšto miško takais. Stebėjimo punktai įrengti aukštuose bokštuose, kuriuose vienu metu gali būti tik vienas girininkas.

Pažymėk mažiausią bokštų skaičių taip, kad būtų galima stebėti visus paveiksle pavaizduotus takus.



Paaiškinimas

Atsakymas:

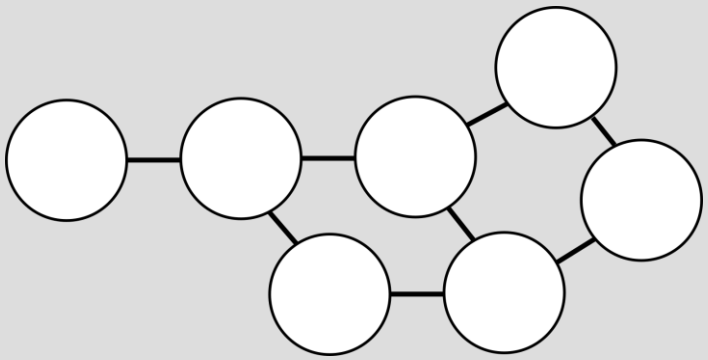


Užtenka 3 bokštų, kad būtų stebimi visi 8 takai.

Yra 8 takai. Jei rastume 2 bokštus, kurių kiekvieną suptų 4 skirtingi takai, tai užtektų dviejų bokštų. Tačiau tokių bokštų nėra. Daugiausia 3 nepasikartojantys takai supa bokštus, todėl reikalingi 3 stebėjimo bokštai.

Tai informatika!

Informatikoje įvairūs sudėtingi objektai dažnai vaizduojami grafais. Grafo viršūnės (apskritimai) vaizduoja objektus (bokštus), o briaunos – ryšius tarp jų. Šio uždavinio grafas atrodo taip:



Šis uždavinys yra analogiškas uždaviniui, kai ieškoma minimalaus skaičiaus gatvių apšvietimo stulpų ar minimalaus stebėjimo kamerų skaičiaus. Šis uždavinys dar vadinamas minimalaus grafo viršūnių padengimo (angl. *vertex cover*) uždaviniu, kai duotame grafe ieškomas mažiausias viršūnių, kurios aprėpia visas grafo briaunas, skaičius.

30. Taškūnai

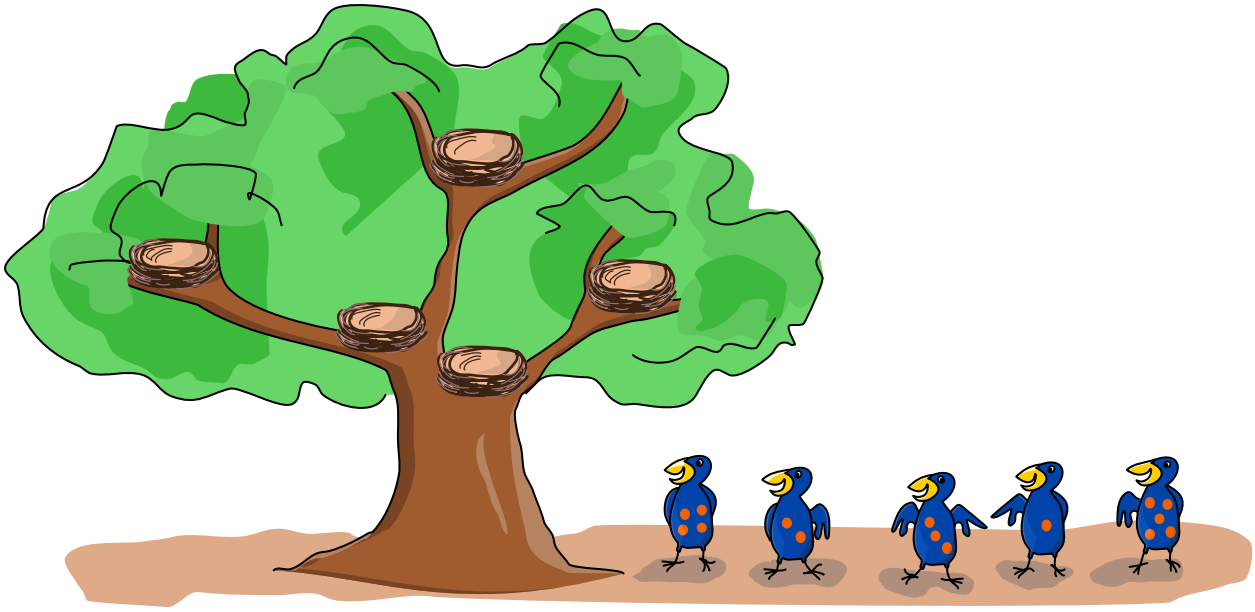
Taškūnai yra taškuoti paukščiai, kurie patys nesuka lizdų, o ieško medžio su apleistais lizdais ir juose apsigyvena.

Lizdo paiešką jie pradeda nuo medžio apačios ir kartoja tolesnius žingsnius tol, kol randa tinkamą lizdą:

1. Kyla į viršų, kol pasiekia lizdą.
2. Jei lizdas tuščias, taškūnas jame apsigyvena.
3. Jei lizdas užimtas ir lizdą užėmusio taškūno taškų skaičius
    - didesnis, tai skrenda į kairę;
    - mažesnis arba toks pat, tai skrenda į dešinę.

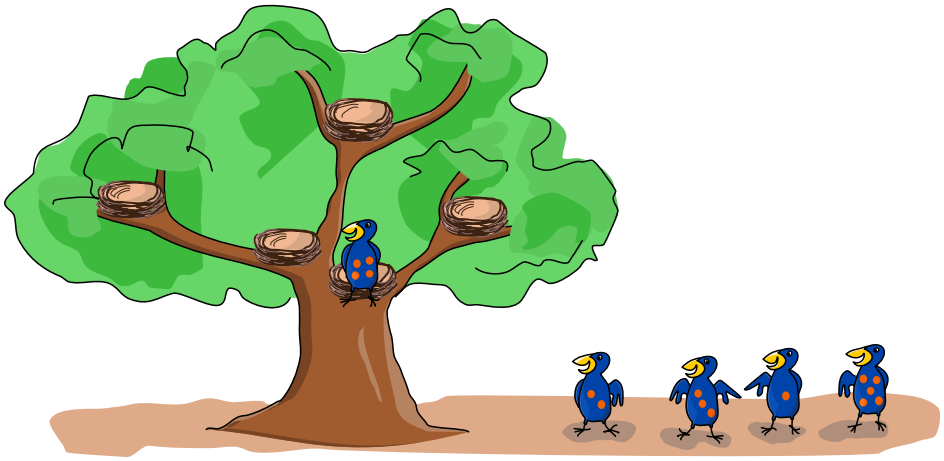
Padėk atskridusiems taškūnams apsigyventi – nuvilk juos į tinkamus lizdus.

Taškūnai lizdus renkasi eilės tvarka, pirmas renkasi esantis arčiausiai medžio.

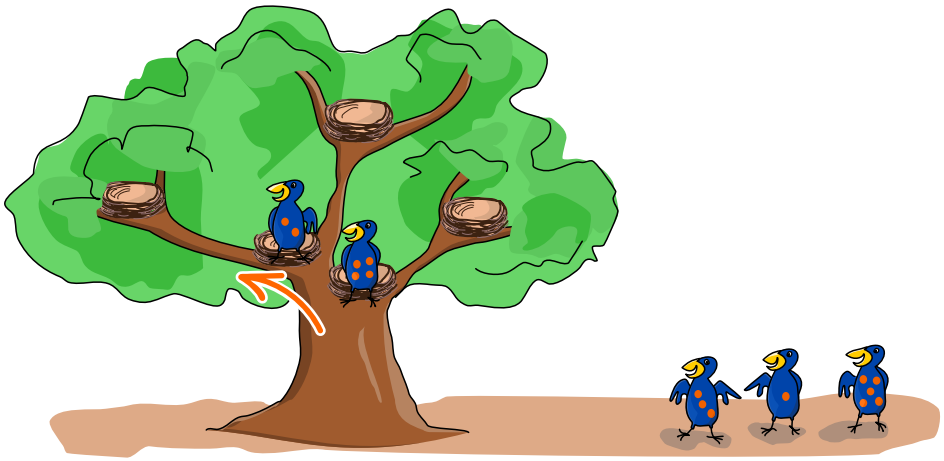


Paaiškinimas

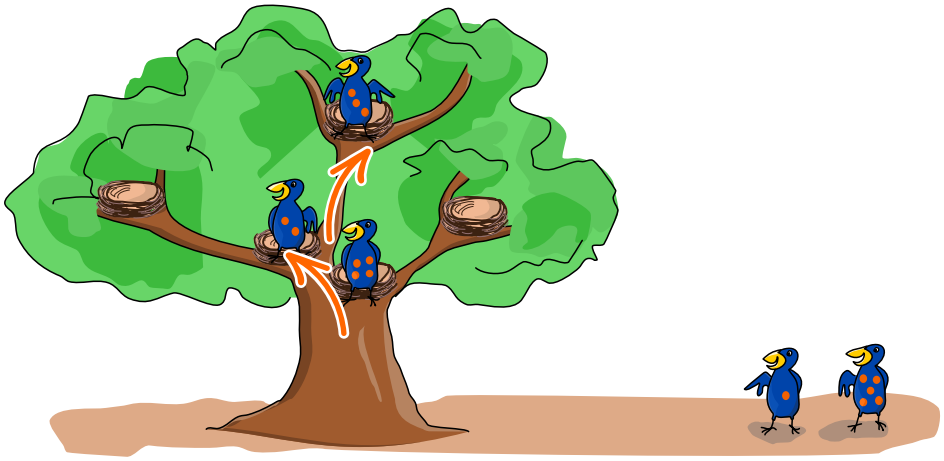
Taškūnas su keturiais taškais apsigyvens žemiausiame lizde.



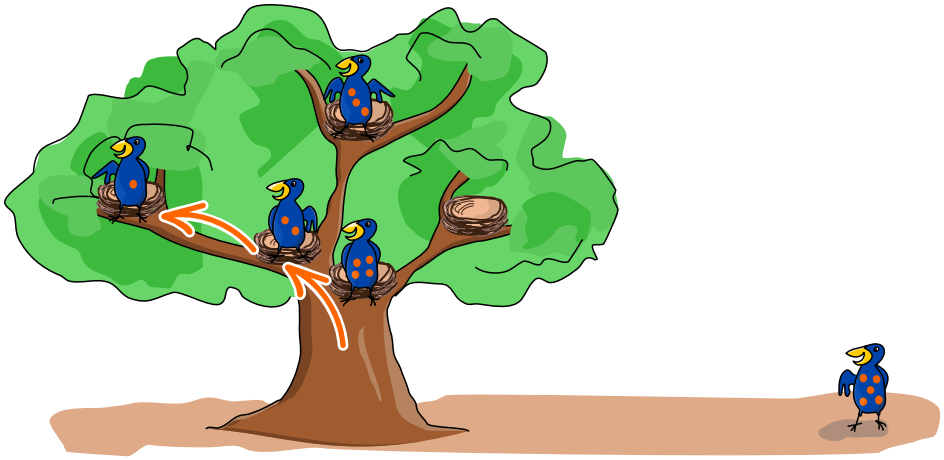
Antrasis taškūnas skris kairėn į artimiausiąjį tuščią lizdą, kadangi pirmame lizde jau tupi pirmasis taškūnas su keturiais taškais.



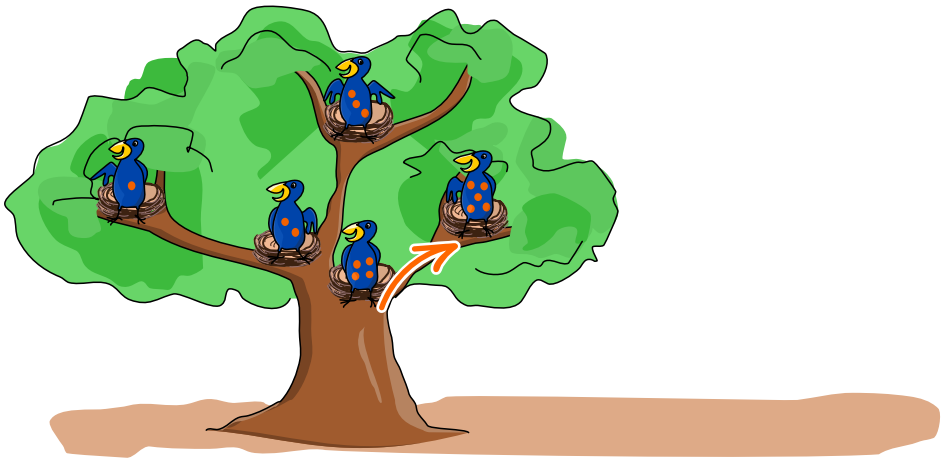
Trečiasis taškūnas skrenda iki artimiausiojo lizdo kairėje (jo trijų taškų skaičius mažesnis už pirmame lizde tupinčio pirmojo taškūno). Toliau jis sutinka taškūną su dviem taškais. Kadangi 2 yra mažiau nei 3, taškūnas toliau skrenda į dešinę, randa tuščią lizdą ir jame apsigyvena.



Ketvirtasis taškūnas vientaškis. Pasižiūri į pirmojo lizdo taškūno taškų skaičių ir skrenda į kairę. Po antrojo lizdo irgi suka į kairę. Kadangi visi sutikti taškūnai turi daugiau taškų, jam teks skristi ir apsigyventi pačiame kairiausiame lizde.



Paskutinis taškūnas penkiataškis. Kadangi jo taškų skaičius didesnis nei tupinčio žemiausiame lizde, tai jis skris į dešinę ir apsigyvens pirmame tuščiame lizde.

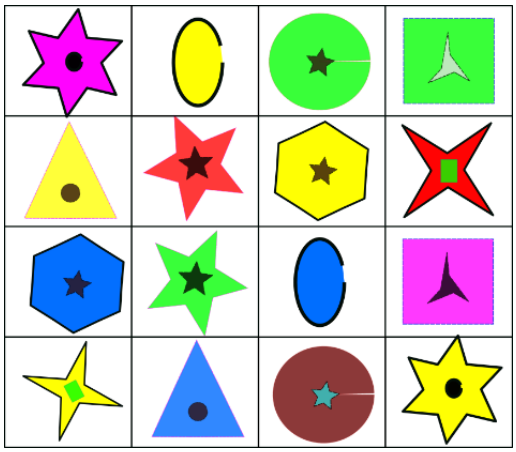


Tai informatika!

Toks duomenų organizavimo metodas vadinamas dvejetainiu paieškos medžiu. Nuo kiekvieno lizdo šakos eina į kitus du lizdus. Dvejetainiai paieškos medžiai naudojami kompiuterinėse programose, ypač kai yra daug duomenų ir juose reikia atlikti paiešką. Kiekviena kartą pridėjus šaką prie dvejetainio medžio, jis didėja. Taigi dvejetainiai medžiai gali keistis, todėl jie vadinami dinamine duomenų struktūra.

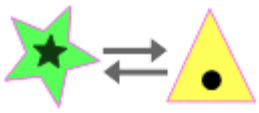
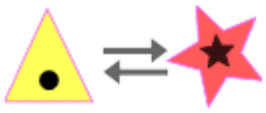
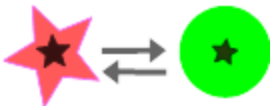
31. Figūrų sukeitimas

Rimtė žaidžia figūrų žaidimą. Žaidimo lenta padalinta į langelius. Pradžioje Rimtė sudėliojo figūras į visus langelius, kaip parodyta paveiksle:



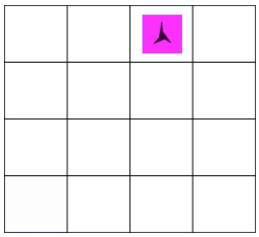
Tada ji sukeitė vietomis figūrų poras, t. y., pakeitė jų pozicijas lentoje.

Rimtė atliko keturis keitimus tokia tvarka:

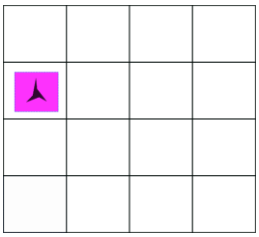
1	2	3	4
			

Kurioje lentos vietoje atsiras  ?

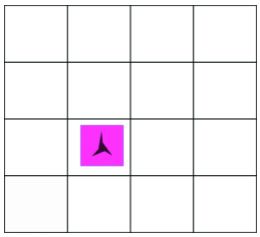
A.



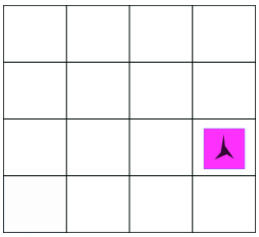
B.



C.



D.

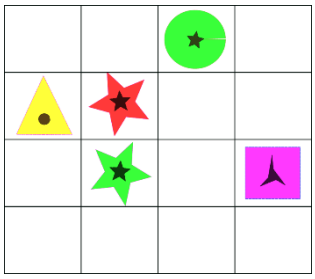


Paaiškinimas

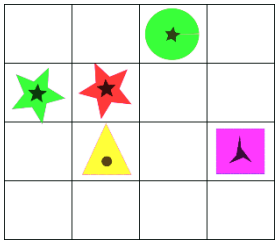
Teisingas atsakymas: C.

Atliekamas tik penkių figūrų keitimas vietomis. Taigi galime nekreipti dėmesio į kitas lentoje esančias figūras.

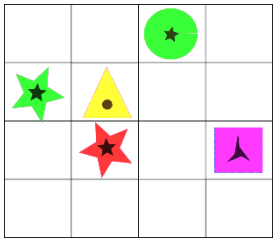
Penkių keitimuose dalyvaujančių figūrų pradinės pozicijos:



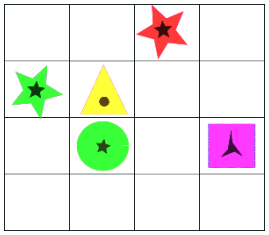
Po 1 sukeitimo



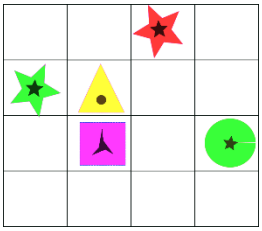
Po 2 sukeitimo



Po 3 sukeitimo



Po 4 sukeitimo



Tai informatika!

Šis uždavinys susijęs su keitimo funkcija. Kintamasis programavime – tai atminties paskyrimas informacijai laikyti. Sukeitimas reiškia dviejų suderinamų pagal duomenų tipą kintamųjų reikšmių sukeitimą.

Pavyzdžiui, jei kintamojo A reikšmė „Vardas“, o kito kintamojo B reikšmė „Pavardė“, tai sukeitus kintamojo A reikšmė bus „Pavardė“, o kintamojo B – „Vardas“.

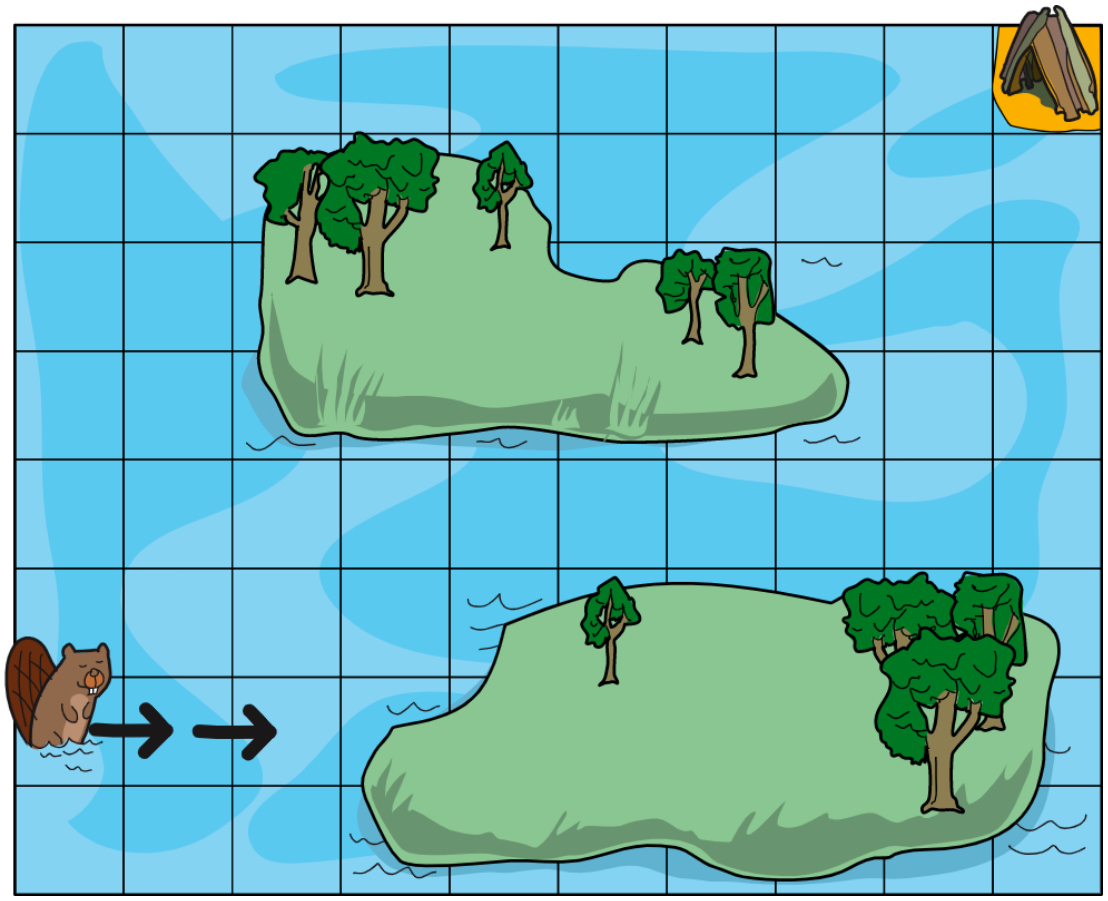
Šie keitimo žingsniai gali būti naudojami rikiuojant duomenų rinkinį tam tikra tvarka, pvz., didėjančiai ar mažėjančiai.

32. Raskite klaidą

Sofija parašė programą, kuri nuvestų bebrą iki jo namo. Programą sudaro tokios 5 komandos:

2→ 2↑ 5→ 4↑ 1→

Pirmoji komanda yra teisinga ir priverčia bebrą judėti taip:



Tačiau viena iš likusių keturių komandų yra klaidinga!

Kurioje komandoje yra klaida?

A. 2↑

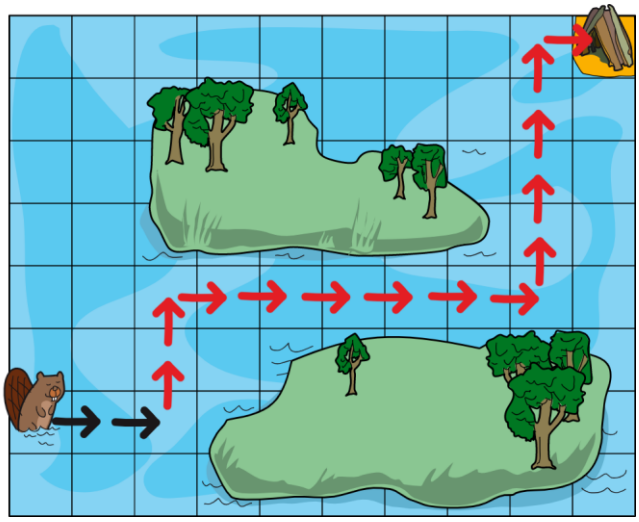
B. 5→

C. 4↑

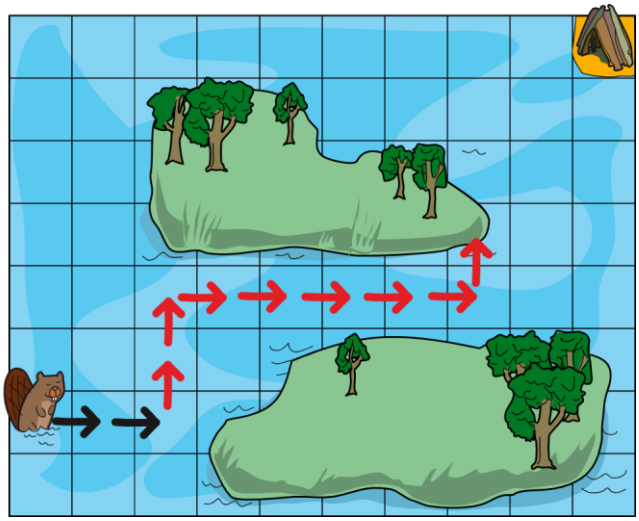
D. 1→

Paaiškinimas

Komanda 5→ turi būti pataisyta į 6→. Tik taip bebras pasieks langelį, kuriame yra jo namas, kaip parodyta paveiksle:



Jei bebras judės pagal originalią Sofijos programoje esančią komandą 5→, jis atsitrenks į salą:



Tai informatika!

Programa sudaryta iš komandų, naudojamų užduočiai išspręsti, ir vaizduoja operacijų ir sprendimų, kurie turi būti atlikti, seką. Pradedame viename taške ir pasiekiame kitą tašką, kuris atitinka teisingą užduoties sprendimą.

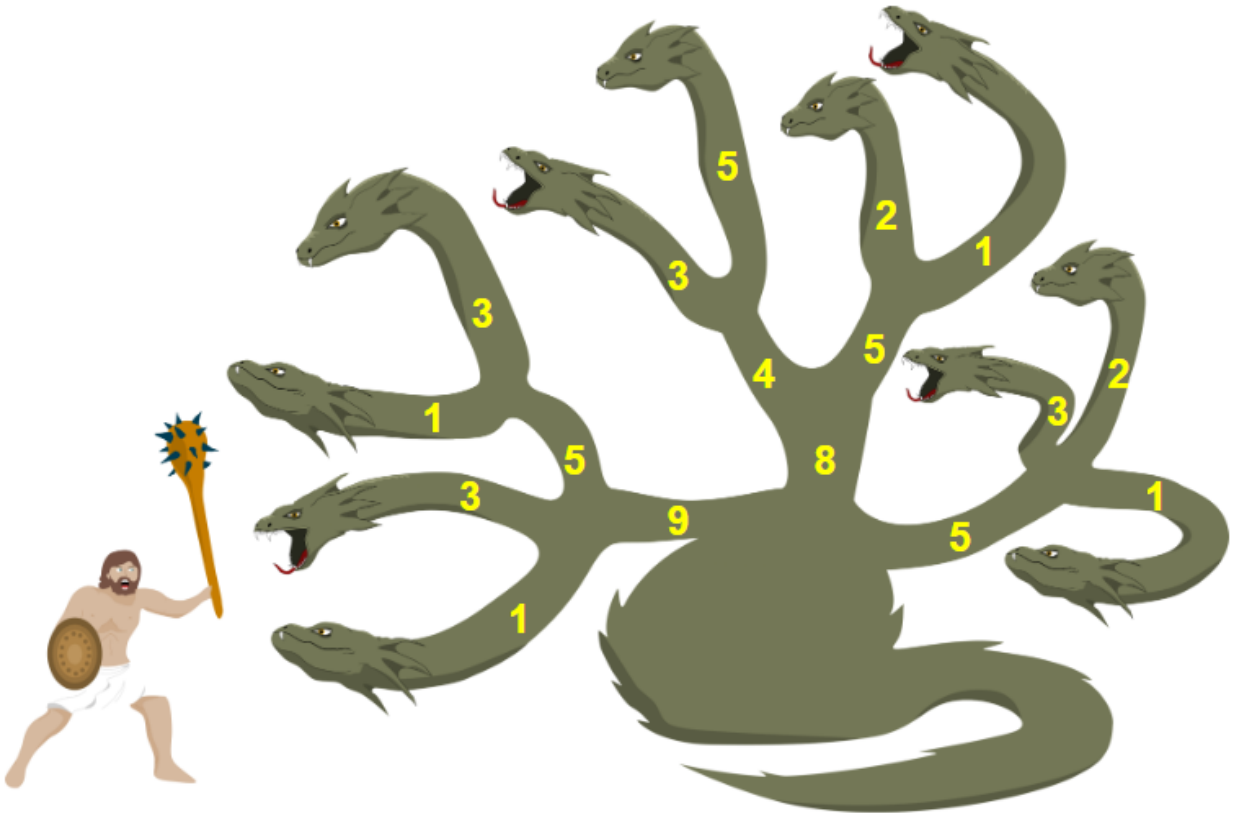
Šioje užduotyje pirmąją komandų seką vaikai gali išsiaiškinti nagrinėdami bebro judėjimą po vieną komandą. Kai bebras atsiranda prie salos, vaikai turi suvokti, kad ankstesnėje komandoje turėjo būti vienu žingsniu daugiau. Verta vaikus paskatinti įvykdyti programą iki galo ir pamatyti, kad bebras baigs judėti nepriėjęs iki savo namo.

Šioje užduotyje mes ieškome klaidos. Tai vadinama programos derinimu, ir šis įgūdis yra svarbus programuotojų darbe.

33. Heraklis ir hidra

Graikų mitologijos devyngalvė jūrų pabaisa hidra niokojo apylinkes, draskė gyvulius. Heraklis stojo į kovą su hidra užsimodamas milžiniška kuoka. Hidra buvo baisiai stipri. Laimei, Heraklis žinojo, kiek kartų reikia suduoti kiekvienai galvai, kad ji netektų galios. Kuoka sunki ir Heraklis, taupydamas jėgas, nori suduoti kiek įmanoma mažiau smūgių, kad visos hidros galvos būtų įveiktos.

Pažymėkite, į kurias vietas Heraklis turi smogti, kad jam reikėtų mažiausiai smūgių įveikti hidrą.



Paaiškinimas

Hidrą galime įsivaizduoti kaip šakotą medį. Tada reikia taikyti minimalių šakų svorius, atskiriant visus lapus nuo šaknies. Tai standartinis grafo minimalaus pjūvių skaičiaus uždavinys, kurį galime išspręsti taikydami įvairius algoritmus (pvz., Fordo ir Fulkesrono algoritmą).

Tačiau galime taikyti daug paprastesnį algoritmą minimaliam pjūvių skaičiui tarp medžio lapų ir šaknies rasti. Pradėkime nuo lapų link šaknies ir perskaičiuokime kiekvieną šaką, ar optimalu ją pjauti, ar ne.

Aptarkime išsamiau. Pradžioje imamės visų šakų, taigi pradinis atsakymas yra 1 + 3 + 1 + 3 + 3 + 5 + 2 + 1 + 3 + 2 + 1.

Toliau judame link šaknies. Kiekvienu tolesniu žingsniu galime palikti „seną“ pjūvį arba pakeisti jį nupjaudami vieną šaką.

Antroji iteracija:

1 + 3 + min(5, 1 + 3) + min(4, 3 + 5) + min(5, 2 + 1) + min(5, 3 + 2 + 1), arba 1 + 3 + 4 + 4 + 3 + 5.

Trečioji iteracija: min(9, 1 + 3 + 4) + min(8, 4 + 3) + 5 arba 8 + 7 + 5.

Štai ir pasiekėme medžio šaknį, gauname galutinį atsakymą 8 + 7 + 5 = 20.

Tai informatika!

Medžiai yra svarbi informatikos duomenų struktūra. Nemažai šiuolaikinių algoritmų naudoja medžius, pavyzdžiui, sprendimų medžiai atsitiktiniame rinkinyje. Šis uždavinys apie minimalų grafo pjūvių skaičių leidžia rasti didžiausią srautą šiame grafe. Tai dažnai naudojama logistikoje, pavyzdžiui, padeda nustatyti maksimalų prekių, kurias galima gabenti iš gamyklų į kitas šalis, svorį atsižvelgus į visas transporto priemones ir jų galimybes.

Grafo minimalų pjūvių skaičių, t. y., suskaidymą į dvi ar daugiau dalių su minimaliomis sąnaudomis, nėra taip lengva apskaičiuoti, kaip parodėme šio uždavinio medžiui, tačiau tai nėra labai sunku. Yra įdomių sprendimo būdų. Gali būti naudojami minimalūs pjūviai, pavyzdžiui, dalijant objektus į panašias dalis. Specialiuose grafuose minimalaus pjūvių skaičiaus išlaidos atitinka didžiausią galimą srautą tinkle.

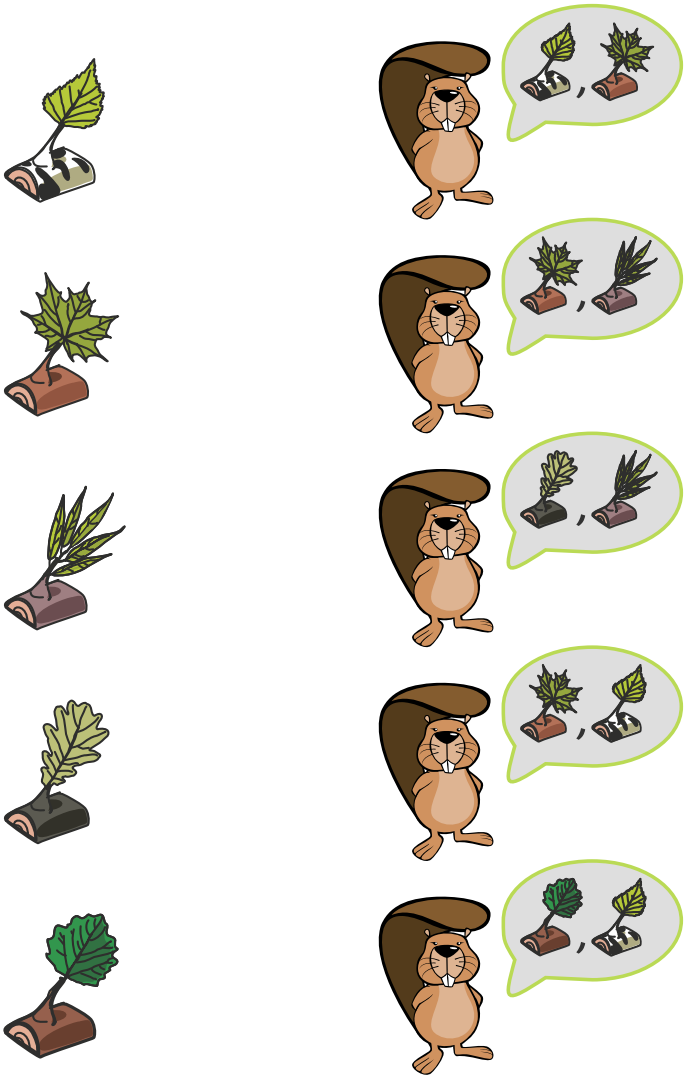
34. Norai

Bebrų šeima turi penkias dovanas, po vieną kiekvienam bebriukui.

Kiekvienas bebriukas yra išsirinkęs dovaną, kurios jis nori labiausiai, ir kitą, jei ta labiausiai pageidaujama jau būtų paimta.

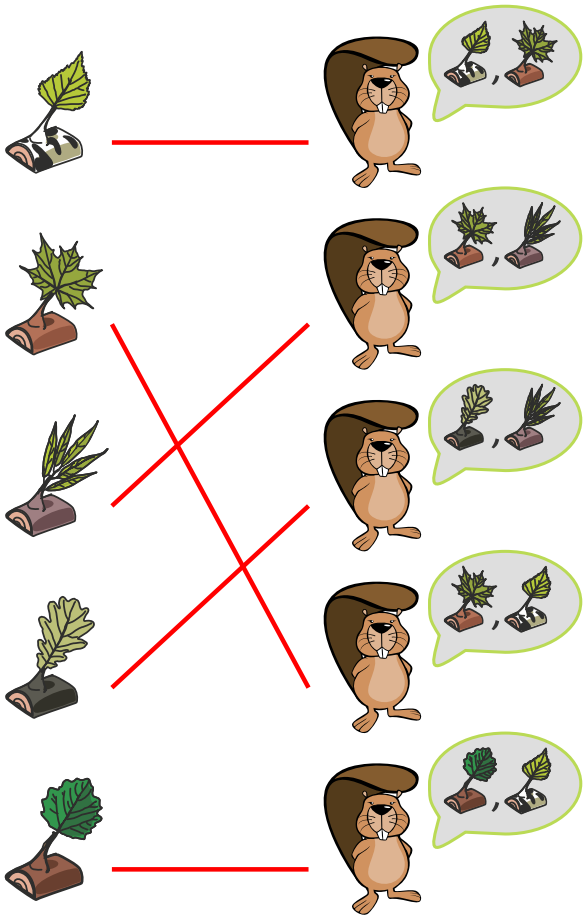
Šeima nori paskirstyti dovanas taip, kad kuo daugiau bebriukų gautų labiausiai norimą dovaną, o kiti gautų antrą pagal pageidavimą.

Įteikite dovanas. Nuvilkite dovanas prie bebriukų.



Paaiškinimas

Paveiksle pavaizduotas dovanų išdalinimas.



Pradžiuginti visus bebriukus jų labiausiai norima dovana neįmanoma, nes du bebriukai labai nori tos pačios dovanos. Parodytame atsakymo paveiksle keturi bebriukai gauna jų labiausiai norimas dovanas, o vienas bebriukas gauna antrą pagal pageidavimą dovaną. Šiai užduočiai geresnis scenarijus neįmanomas. Uždavinį sprendžiant iš viršaus į apačią, antra dovana yra priskiriama antram bebriukui (kaip ir pirma dovana priskiriama pirmam bebriukui), trečia – trečiam, tad ketvirtas bebriukas nebegali gauti labiausiai geidžiamos dovanos. Tai yra vienintelis galimas sprendimas.

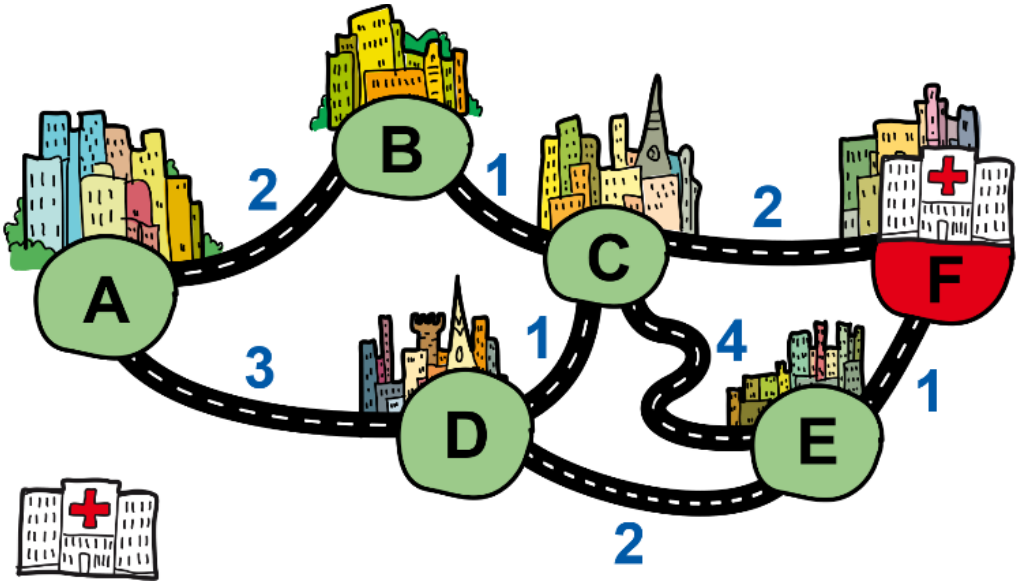
Tai informatika!

Bebriukai surikiavo dovanas pagal savo norus. Kiekvienam bebriukui galima laikyti, kad visos nepaminėtos dovanos pasidalins trečiąją vietą. Bebrų šeima nori išdalinti dovanas taip, kad kuo daugiau bebriukų liktų patenkinti. Dovanų bebriukams suderinimas yra optimalus, jei didžiausiam skaičiui bebriukų bus patenkinti pirmo lygio ir atitinkamai antro lygio norai ir liks mažiausiai nepatenkintų bebriukų. Informatikoje toks derinimas vadinamas maksimalaus lygio derinimu. Pasitaiko labai daug tokio tipo derinimo uždavinių.

35. Ligoninės

Regione yra šeši miestai. Ligoninė gali būti tik dviejuose miestuose. Kai žmonės negaluoja, jie turi kuo greičiau patekti į ligoninę. Žemėlapyje pavaizduoti miestai ir nurodytas laikas valandomis nuvykti iš vieno miesto į kitą.

Viena ligoninė yra F mieste. Nutempkite ligoninės paveiksliuką į tą miestą, kuriame turi būti antroji ligoninė.



Paaiškinimas

Teisingas atsakymas: B miestas.

Kadangi viena ligoninė yra F mieste, tai yra penkios galimybės pasirinkti vietą antrai ligoninei. Kiekvienai galimybei turime patikrinti minimalų laiką, kurio reikia patekti iš kiekvieno miesto į bet kurią ligoninę. Duotos lentelės stulpeliuose rodoma kiekviena galima kiekvienos iš dviejų ligoninių vieta, o eilutėse – skaičius valandų, per kurias galima pasiekti artimiausią ligoninę.

Minimalus laikas vykti iš miesto...	Ligoninių vietos dviejuose miestuose				
	A ir F	B ir F	C ir F	D ir F	E ir F
A	0	2	3	3	5
B	2	0	1	2	3
C	2	1	0	1	2
D	3	2	1	0	2
E	1	1	1	1	0
F	0	0	0	0	0
Minimumas	3	2	3	3	5

Kaip matyti paskutinėje lentelės eilutėje, mažiausias skaičius valandų, reikalingų pasiekti ligoninę, yra antruoju atveju, kai ligoninė yra B mieste. Tai yra geriausias sprendimas – žmonės iš kiekvieno miesto galėtų pasiekti ligoninę ne daugiau kaip per 2 valandas.

Tai informatika!

Ši užduotis yra susijusi su vadinamuoju viršūnių k-centro uždaviniu, kurį galima išspręsti naudojant kompiuterių mokslininkų sukurtą algoritmą. Tai gali būti vaizduojama grafu, t. y., rinkiniu viršūnių, kurias galima sujungti briaunomis, turinčiomis svorį (miestai ir kelionės trukmė tarp jų šioje užduotyje).

Ši problema yra labai dažna, kai bandoma pasirinkti vietas, kur statyti visuomeninės paskirties objektus, pavyzdžiui, mokyklas, ligonines, gaisrines, policijos nuovadas ir kt. Parametras, kuris turėtų būti minimizuojamas, gali būti laikas, per kurį reikia pasiekti objektus, kelionės atstumas ar bet koks kitas turimas kriterijus.

36. Katino nuotraukos

Austėjai patinka fotografuoti savo katiną ir publikuoti jo nuotraukas Bebragrame.

Norėdama sukurti vaizdo įrašą iš katino pirmų trejų metų nuotraukų, Austėja turi parsisiųsti visas nuotraukas į savo kompiuterį ir tinkamai jas įvardyti.

Kompiuteris išdėsto nuotraukas eilės tvarka pagal jų pavadinimus nuo A iki Ž ir nuo 0 iki 9. Austėja nori, kad nuotraukos būtų išdėstytos nuo seniausios iki naujausios, todėl ji prideda datą prie kiekvienos nuotraukos pavadinimo.

Kurį nuotraukos pavadinimo šabloną turėtų naudoti Austėja?

A) katinas_19_rugpjūčio_2021	E) katinas_2021_rugpjūčio_19
B) katinas_rugpjūčio_19_2021	F) katinas_2021_19_8
C) katinas_19_8_2021	G) katinas_2021_08_19
D) katinas_19_08_2021	H) katinas_2021_8_19

Paaiškinimas

Austėja turėtų rinktis G šabloną.

A, B, C ir D atsakymai yra neteisingi.

Jei turime skirtingų metų nuotraukas, reikia, kad visos tų pačių metų nuotraukos būtų išdėstytos šalia, net jei jų mėnesiai ir dienos skiriasi. Todėl metai varde turi eiti prieš mėnesį ir dieną. Tuomet tų pačių metų nuotraukos turės vardą su vienoda pradžia ir kompiuteris jas išdėstys greta.

Jei mėnuo ar diena būtų prieš metus (A, B, C ir D šablonai), tai to paties mėnesio ir dienos, bet skirtingų metų nuotraukos būtų išdėstytos greta. Pavyzdžiui, rugpjūčio 19 d. nuotrauka, padaryta 2022 m., būtų šalia rugpjūčio 19 d. nuotraukos, padarytos 2021 m., kas neatitinka Austėjos tikslo.

F – neteisingas atsakymas. Dėl tos pačios priežasties mėnuo turi būti nurodomas prieš dieną: jei ir metai, ir mėnuo yra vienodi, nuotraukos bus išdėstytos šalia. Dėl to F šablonas nesuveiks: juo naudojantis 2021 m. spalio 19 d. nuotrauka būtų surikiuota tarp tų pačių metų rugpjūčio 18 d. ir rugpjūčio 20 d.

E – neteisingas atsakymas. Matome, kad mėnuo turėtų būti rašomas skaitiniu pavidalu. Jei rašytume žodžiu, kaip yra E šablone, tai, pavyzdžiui, balandžio ir birželio nuotraukos atsirastų greta, nors šalia balandžio mėn. nuotraukų turėtų būti gegužės mėn. nuotraukos, o ne birželio.

H – neteisingas atsakymas. Pagrindinis skirtumas tarp G ir H šablonų yra tas, kad G variante mėnuo užrašomas su nuliu priekyje: „08“, o H variante – be nulio: „8“. Mėnesiams nuo sausio iki rugsėjo abiejų šablonų schema puikiai veiks. Tačiau spaliui, lapkričiui ir gruodžiui, turintiems du skaitmenis savo skaitiniame pavidale, rikiavimas neveiks tinkamai, jei pasirinksime H šabloną.

Pavyzdžiui, 2021 m. rugpjūčio 19 d. ir 2021 m. gruodžio 19 d. nuotraukos. Jei būtume pasirinkę H šabloną, šios datos būtų užrašomos 2021_8_19 ir 2021_12_19. Norėtume, kad 2021_8_19 eitų prieš 2021_12_19, tačiau kompiuteris pirmoje vietoje išdėstys 2021_12_19, kadangi „1“ (mėnesio „12“ pirmas skaitmuo) eina prieš „8“.

Tai informatika!

Duomenų išdėstymas tam tikra tvarka informatikoje vadinamas rikiavimu.

Rikiavimas labai svarbus, kadangi naudojamas daugelyje kitų algoritmų. Taigi rikiavimo efektyvumas veikia kitų algoritmų efektyvumą. Rikiavimo algoritmo pasirinkimas priklauso nuo norimų rikiuoti duomenų. Paprasčiausi rikiavimo algoritmai, kurie Jums jau gali būti žinomi, yra burbulo, įterpimo ir pasirinkimo rikiavimo algoritmai.

Datų pateikimas failuose ir kompiuterio atmintinėje yra svarbus informatikos uždavinys. Iki 2000 metų daugelis kompiuterių sistemų naudojo tik du skaitmenis metams užrašyti, pavyzdžiui, 81 atitiko 1981 metus. Dabar dauguma kompiuterių sistemų laikosi šablono YYYYMMDD. Čia YYYY reiškia keturis metų skaitmenis, MM yra du mėnesio skaitmenys, o DD atitinka dienos du skaitmenis. Šis šablonas veiks kompiuteriuose dar maždaug 8 000 metų nuo dabar, t. y., iki 10000 metų.

37. Dovana

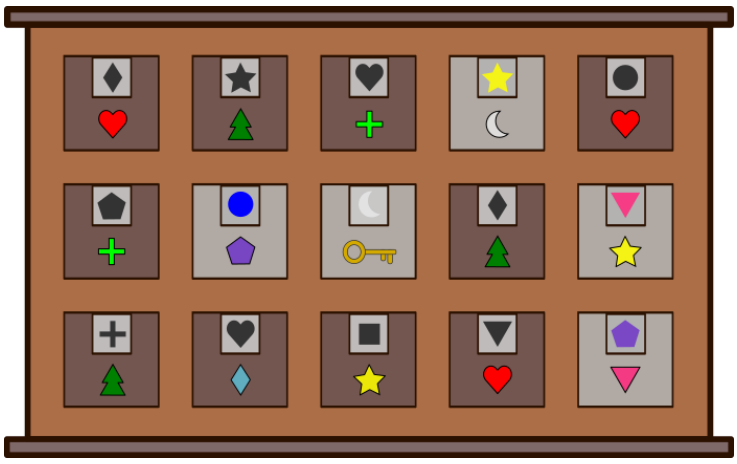
Barboros mama nupirko dovaną ir ją užrakino seife. Ji padavė Barborai mėlyną rutulį ir pasakė: „Dovaną galėsi gauti, jei sugebėsi išspręsti galvosūkį ir paimti raktą, kuris yra viduriniame stalčiuje.“

Norėdama atidaryti stalčių, Barbora į stalčiaus rakto skylutę (stalčiaus viršuje) turi įdėti atitinkamos formos daiktą. Tuomet stalčius atsidaro ir ji gali paimti daiktą, kuris guli stalčiuje ir yra nupieštas stalčiaus priekyje (apatinėje dalyje).

Pasirinkite tvarką, pagal kurią spausdami galimus atidaryti stalčius, gausite raktą.

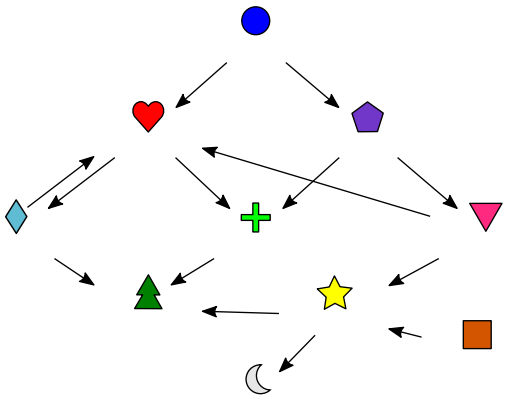
Paaiškinimas

Teisingas atsakymas:



Užduotis yra surasti kelią tarp rutulio (skritulio) ir mėnulio, nes rakto skylutė ant stalčiaus, kuriame yra seifo raktas, yra mėnulio formos.

Sprendimą rasti gerokai lengviau nusipiešus schemą, kurioje pavaizduoti objektai ir ryšiai tarp jų. Iš kiekvieno objekto į kitą eina rodyklės (orientuotas grafas), kurios parodo, ką galima rasti atitinkame stalčiuje.



Galima naudoti ir atvirkštinį kelią – nuo mėnulio iki rutulio (skritulio), kad rastumėme teisingą sprendimą.

Tai informatika!

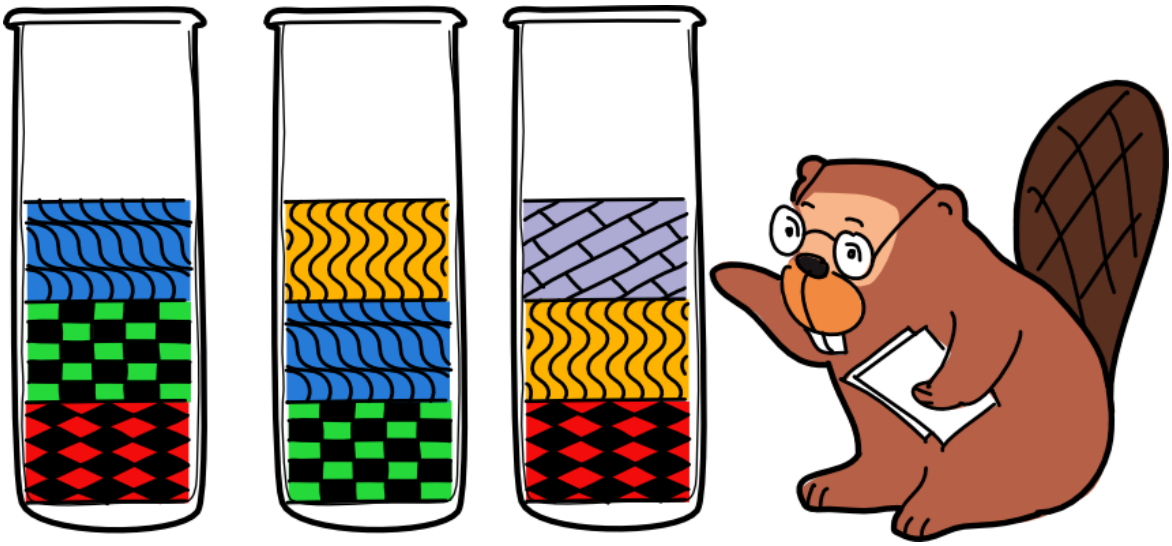
Grafas yra diagrama arba vaizdinis sąryšių vaizdavimas tarp dviejų ar daugiau objektų.

Objektai ir stalčiai apibūdina orientuotą grafą: kiekvienas objektas yra viršūnė, sujungta briaunomis, vaizduojančiomis ryšius tarp objektų ir to, koks rakto skylute pavaizduotas objektas leidžia paimti kokį objektą iš stalčiaus. Informacijos vaizdavimas grafu leidžia pamatyti uždavinio struktūrą.

Galima naudoti ir atgalinį kelią grafe, norint surasti sprendimą. Taip pat labai svarbu sugebėti atpažinti šablonus ir figūras.

38. Skysčiai

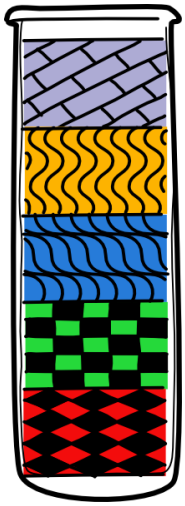
Marius nori nustebinti savo draugą, kurio gimtadienis rytoj. Jis nori padovanoti siurprizą – buteliuke susluoksniuotą įvairių rūšių skystį. Mokytoja paaiškino, kad skystis, kurio tankis didesnis, bus žemiau mažesnio tankio skysčių. Mokytoja padavė Mariui 3 buteliukus su susluoksniuotais laboratorijoje turimų spalvotų skysčių pavyzdžiais:



Marius ketina padovanoti buteliuką su 5 skysčių sluoksniais. Supilstyk (nutempk) skysčius teisingai.

Paaiškinimas

Skysčių tvarka, remiantis jų tankiu, yra tokia:



Tai informatika!

Spręsdami šią užduotį mokiniai, matydami pavyzdžiuose pasikartojančius raštus, turėtų nustatyti komponentų panašumus (atpažinti šablonus). Šablonų atpažinimas reiškia paprastą šablonų paiešką užduotyje ir bet kokių sprendimų, išmoktų praeityje, panaudojimą naujoje situacijoje. Praeityje išmokti dalykai gali mums padėti išspręsti naujas problemas.

Rikiavimas yra daiktų padėjimas į teisingą vietą remiantis kokia nors taisykle. Mūsų aptariamo uždavinio skysčių rikiavimo taisyklę apibrėžia kiekvieno skysčio tankis.

Uždavinio sprendinys turi tenkinti sąlygą, vadinamą ribojimu. Šiuo atveju skysčių tankis tampa ribojimu rikiuojant skysčius.

39. Reguliuojami stalai

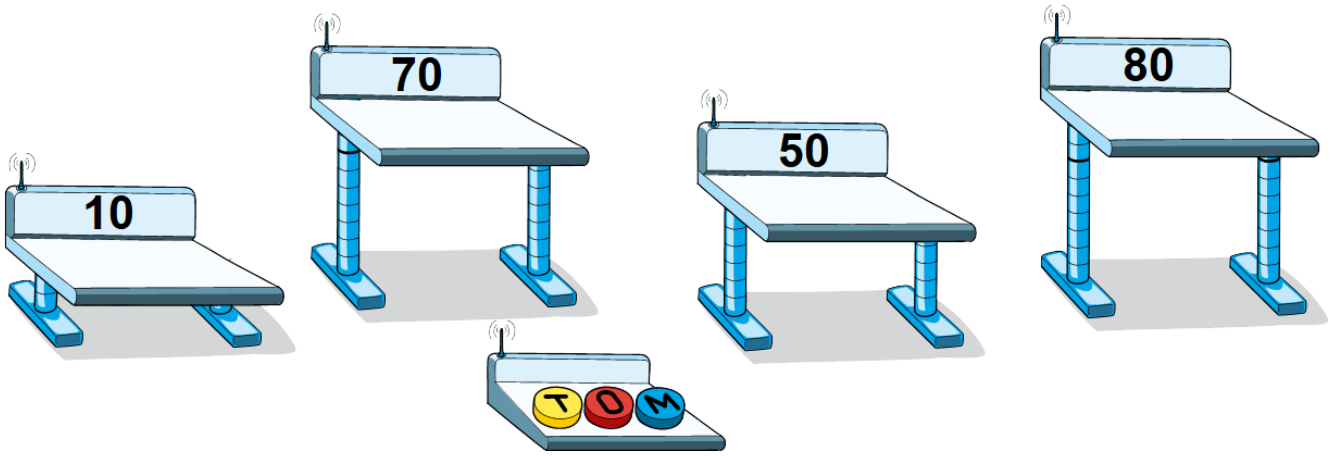
Modernioje klasėje stovi automatiniai stalai, kurių aukštį galima reguliuoti. Prasidedant pamokoms visi stalai turi būti 60 cm aukščio. Stalų aukštis keičiamas naudojant nuotolinio valdymo pulto mygtukus T, O ir M.

Prieš pamokas kažkas žaidė su nuotolinio valdymo pultu ir jį perprogramavo. Dabar pulto mygtukai veikia taip:

- T mygtukas padidina 1, 2 ir 3 stalų aukščius 10 cm;
- O mygtukas sumažina 2, 3 ir 4 stalų aukščius 10 cm;
- M mygtukas padidina 1, 3 ir 4 stalų aukščius 10 cm.

Šie veiksmai atliekami kiekvieną kartą paspaudus mygtuką.

Šiuo metu visų keturių stalų aukščiai 10 cm, 70 cm, 50 cm ir 80 cm.



Kaip reikėtų spaudyti mygtukus, kad visų keturių stalų aukštis būtų 60 cm?

a) T mygtuką paspausti 4 kartus, O – 5 kartus ir M – 1 kartą.

b) T mygtuką paspausti 5 kartus, O – 1 kartą ir M – 0 kartų (nespausti).

c) T mygtuką paspausti 3 kartus, O – 4 kartus ir M – 2 kartus.

d) T mygtuką paspausti 2 kartus, O – 4 kartus ir M – 6 kartus.

Paaiškinimas

Teisingas atsakymas: c.

Reikia 3 kartus nuspausti T mygtuką, 4 kartus O mygtuką ir 2 kartus M mygtuką.

Pastebėsime, kad nuotolinio valdymo pultas veikia taip, kad paspaudus kiekvieną mygtuką stalų aukštis pasikeičia 10 cm, t. y., visada tuo pačiu skaičiumi. Du mygtukai (T ir M) paaukština atitinkamus stalus, o vienas mygtukas (O) pažemina atitinkamus stalus. Be to, kiekvienas mygtukas keičia tik trijų stalų aukščius, todėl visada bus vienas stalas, kurio aukštis nekis.

1-as stalas yra 50 cm per žemas. Iš to darome išvadą, kad arba T mygtuką, arba M mygtuką turime paspausti lygiai 5 kartus (arba ir vieną, ir kitą, bet iš viso turi būti 5 paspaudimai). O mygtukas nedaro jokio poveikio 1-am stalui, todėl juo negalime šio stalo paaukštinti. Šitaip samprotaudami galime atmesti *d* atsakymą, nes T + M = 8. Paspaudus *d* sprendime išvardintus mygtukus 1-o stalo aukštis būtų 10 + 20 + 60 = 90 cm (pradinis aukštis + 2 x 10 paspaudus T mygtuką + 6 x 10 paspaudus M mygtuką).

2-as stalas yra 10 cm per aukštas. M mygtukas 2-am stalui nedaro įtakos. Taigi T + O = -1. Šitaip samprotaudami galime atmesti *b* atsakymą (2-o stalo aukštis tada būtų 70 + 50 - 10 = 110).

3-as stalas yra 10 cm per žemas, todėl gauname, kad T - O + M = 1. Tai reiškia, kad *a* ir *b* atsakymai gali būti atmesti: *a* atveju 3-o stalo aukštis nesikeistų (50 + 40 - 50 + 10 = 50); *b* atveju 3-o stalo aukštis būtų 50 + 50 - 10 = 90.

Tad visi variantai atmesti, išskyrus *c*.

Tačiau dar reikia patikrinti, ar *c* variantas tinka 4-am stalui. 4-as stalas yra 30 cm per aukštas, T mygtukas neturi įtakos šio stalo aukščiui. Taigi O mygtuką reikia paspausti 3 kartus ir kiekvieną kartą paspaudus M mygtuką vėl spausti O mygtuką. Remiantis *c* varianto mygtukų paspaudimų seka, 4-o stalo aukštis bus 80 - 40 + 20 = 60.

Kadangi jau įsitikinome, kad *c* atsakymas tinka 1, 2 ir 3-am stalams, tad esame tikri, kad šis variantas geras.

Kitaip sprendimą galima rasti sudarius keturias tiesines lygtis. Kiekvienam stalui sudarykime lygtį, kurie mygtukai keičia stalo aukštį ir kokio aukščio pokyčio ieškoma. Pavyzdžiui, 1-o stalo aukštį galima keisti tik mygtukais T ir M, o norimas aukščio pokytis yra 50 cm, kurį galima pasiekti 5-ais mygtukų paspaudimais (nes vienu mygtuko paspaudimu pakeliame 10 cm).

Kadangi yra keturi stalai ir trys mygtukai, reikia sudaryti keturias tiesines lygtys su trimis nežinomaisiais:

T + M = 5 T – O = -1 T – O + M = 1 - O + M = -2	}	Jei iš pirmosios atimsime trečiąją lygtį, gausime O = 4. Įstatę į antrąją lygtį, gausime T = 3. Tik kai M = 2, visos lygtys tenkinamos. Vadinasi, tai yra vienintelis sprendimas.

Tai informatika!

Tai yra tipinis diskrečiosios matematikos optimizavimo uždavinys (tiesinio programavimo uždavinys). Šiam uždaviniui numatyti ribojimai. Toks tikslo siekimas būdingas informatikai. Ieškoma veiksmų sekos, vedančios į tam tikrą tikslą.

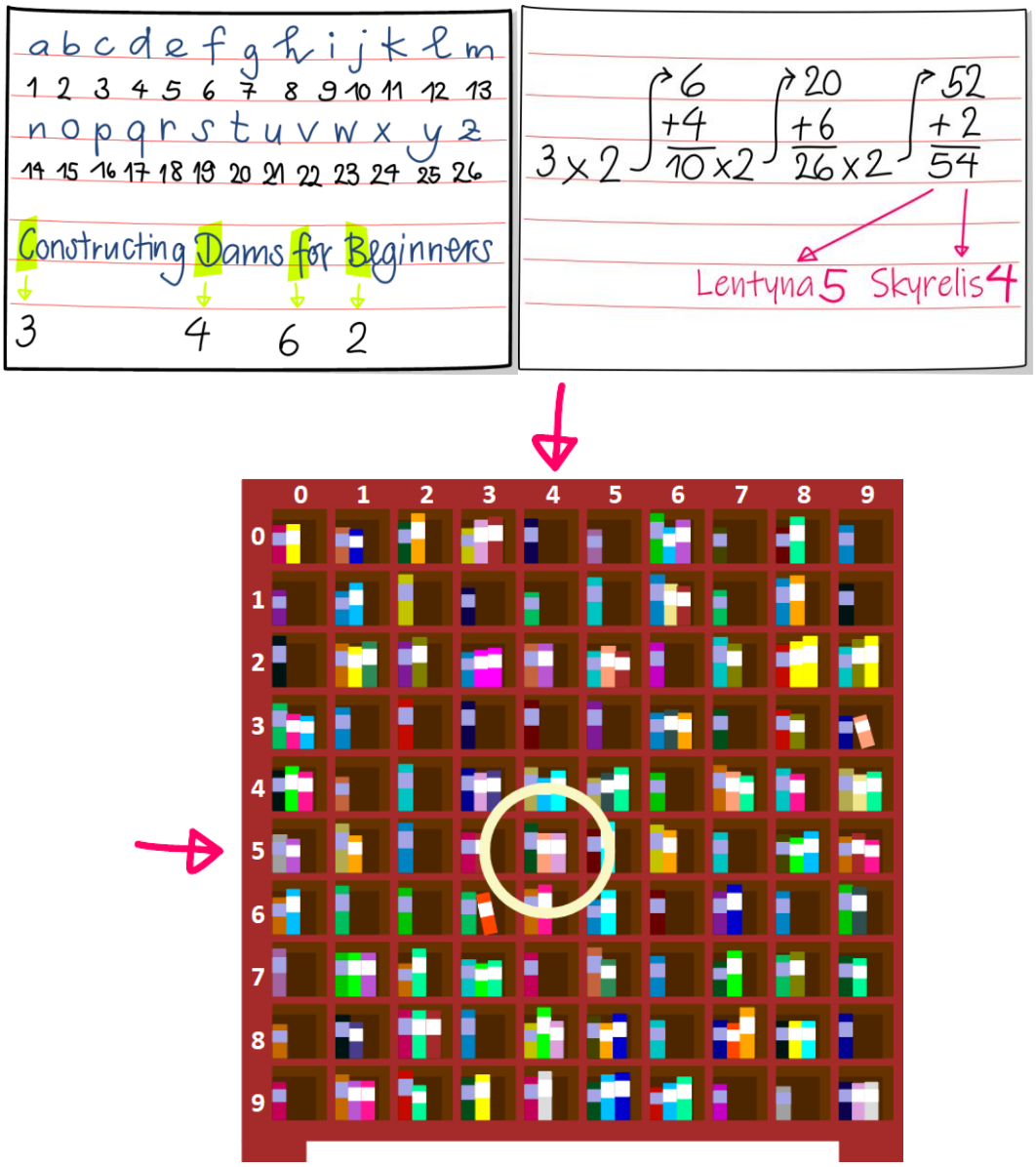
Visą uždavinį galima apibūdinti net kaip kelio paiešką keturių matmenų erdvėje esant trims leistiniems judėjimo veiksmams, būtent, judant iš taško (10,70,50,80) iki taško (60,60,60,60).

Šiame pavyzdyje galimas tik vienas sprendimas, tačiau paprastai tokie uždaviniai turi daug sprendimų, kuriuos reikia optimizuoti siekiant tikslo. Tada reikia ieškoti tiesinės funkcijos T + M + O minimumo.

40. Knyga

Tomas ir jo vyresnioji sesuo Saulė eina į viešąją biblioteką. Bibliotekoje yra tik viena didelė knygų spinta. Tomas ir Saulė nori pasiskolinti knygą „Constructing Dams for Beginners“.

Atėjusi į biblioteką, Saulė eina tiesiai prie knygų spintos ir išsitraukia norimą knygą. „Iš kur tu žinojai, kurioje vietoje yra knyga?“ – nustebęs klausia Tomas. Saulė nusišypso ir parodo jam du popieriaus lapus:

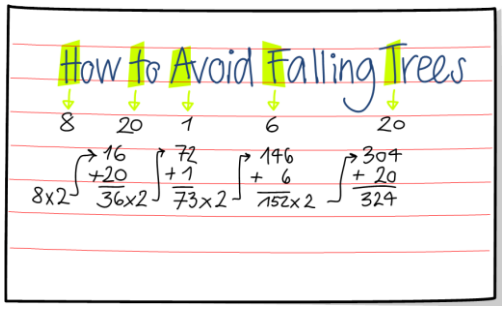


„Aš paėmiau knygos pavadinimo visų žodžių pirmąsias raides ir priskyriau joms po skaičių iš lentelės su lotyniškų raidžių eilės numeriais. Paėmiau pirmos raidės eilės numerį. Prieš pridėdama antros raidės eilės numerį ankstesnį rezultatą padauginau iš 2. Taip skaičiavau, kol pridėjau paskutinės raidės eilės numerį. Gauto rezultato priešpaskutinis skaitmuo parodė man lentynos numerį, o paskutinis skaitmuo – skyrelio numerį“ – paaiškino Saulė. „O ką daryti, jei rezultatas būtų didesnis nei 99?“ – paklausė Tomas. Saulė atsakė: „Reikia tiesiog ignoruoti visus skaitmenis, išskyrus du paskutinius“.

Kur galite rasti knygą „How to Avoid Falling Trees“?

Paaiškinimas

Teisingas atsakymas: knyga yra 2-os lentynos 4-ame skyrelyje. Apskaičiuojama taip:



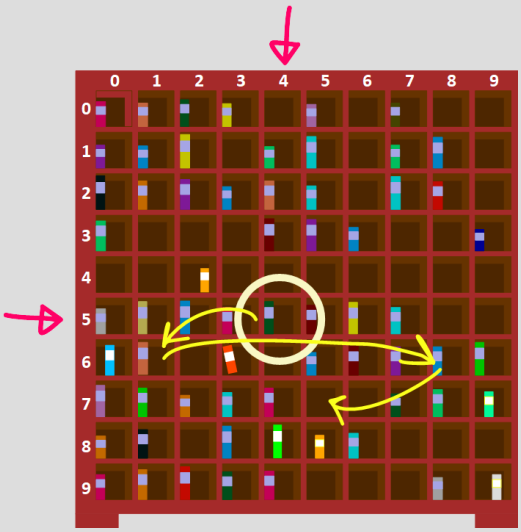
Tai informatika!

Bebrų viešosios bibliotekos sistema remiasi maišos (angl. *hashing*) algoritmo koncepcija.

Jeigu spintoje knygos būtų išdėstytos be jokios sistemos, reikėtų nuosekliai peržiūrėti visas knygas. Norimai knygai rasti tektų patikrinti vidutiniškai 50 proc. visų knygų. Įsivaizduokime, kiek laiko prireiktų ieškant knygos Aleksandrijos bibliotekoje (~100000 knygų), JAV Kongreso bibliotekoje (~38000000 knygų) ar net Jūsų vietinėje viešojoje ar mokyklos bibliotekoje.

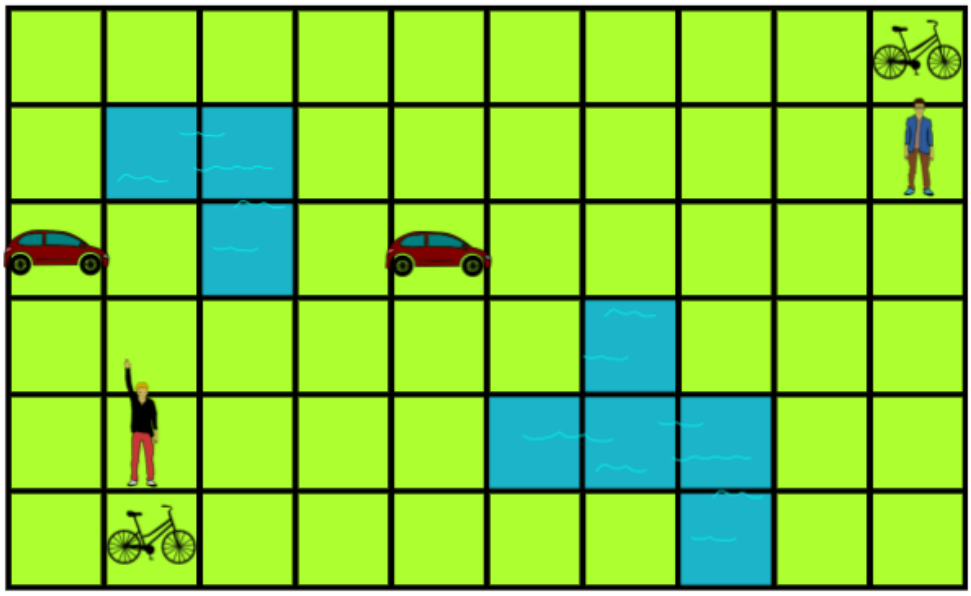
Tokia problema egzistuoja ne tik bibliotekose. Didelėse vaistinėse taip pat reikalinga sistema vaistams laikyti ir surasti. Pastaruoju metu vis daugiau vaistinių pritaiko automatines saugojimo sistemas. Tokioms sistemoms nėra svarbu, kad vaistai būtų laikomi susisteminti tam tikra tvarka, pavyzdžiui, pagal vaistų tipą. Tačiau joms svarbu, kad vaistų spintos būtų tolygiai užpildytos.

Kaip tik čia ir pritaikomas maišos algoritmas. Maišos kodas yra skaičius, kurį apskaičiuoja maišos funkcija, remdamasi daikto ypatybėmis. Mūsų uždavinio atveju knygos pavadinimas transformuojamas į du skaitmenis, kurie nurodo vietą spintoje, t. y., lentyną ir skyrelį. Aišku, gali atsitikti taip, kad skirtingi pavadinimai transformuojami į vienodus maišos kodus, pavyzdžiui, knygos „Tree Bark Gourmet Guide“ ir „Tasty Trees to Gnaw On“. Tokį konfliktą išspręsti galima trimis skirtingais būdais. Vienas būdas – paprasčiausiai padėti kelis daiktus į tą pačią vietą, pvz., lentynos skyrelį. Jei tai neįmanoma, tuomet pasirenkama šalia esanti tuščia vieta arba tuščia vieta, esanti už n vietų, kad užpildymas būtų kuo tolygesnis. Kai ieškoma norimo daikto, paprasčiausiai patikrinamos per tą patį skaičių nutolusios vietos, kol bus rasta tuščia vieta. Kad ilgainiui būtų užpildytos visos vietos, skaičius n yra parenkamas toks, kad jis ir visų vietų skaičius būtų tarpusavy pirminiai skaičiai (pvz., čia pateiktame paveiksliuke parinktas skaičius 7).



41. Skubus pasimatymas

Du draugai norėtų kuo greičiau susitikti. Jie gali pereiti į gretimą kairėje, dešinėje, viršuje arba apačioje langelį. Pėsčiomis tai užtruks 1 minutę. Jei jie patenka į langelį su transporto priemone, gali ja pasinaudoti. Dviračiu per vieną minutę galima įveikti 2 langelius, o automobiliu – 5 langelius. Vandens telkinių kirsti negalima.



Kiek mažiausiai minučių reikia, kad draugai susitiktų viename langelyje?

Paaiškinimas

Teisingas atsakymas: 4. Draugai gali susitikti už 4 minučių.

Reikia įsitikinti, kad 3 minučių nepakanka, kad draugai susitiktų. Nors per 3 minutes galima pasiekti kairėje esantį automobilį, tačiau nebelieka laiko kur nors juo nuvažiuoti. Kitas draugas reikiamo langelio irgi negali pasiekti per 3 minutes. Taigi automobiliai nepadėtų susitikti per 3 minutes. Du draugus vieną nuo kito skiria daugiau kaip 5 minutės, kad susitiktų eidami pėsčiomis, todėl lieka panagrinėti dviračio variantą. Akivaizdu, kad abu turėtų važiuoti dviračiais, kadangi draugus skiria daugiau kaip 9 langeliai. Tačiau dviračiui pasiekti reikia vienos minutės, o per likusias dvi minutes jie negali įveikti likusių langelių net dviračiais.

Tai informatika!

Kaip sprendėte šį uždavinį? Ar atsitiktinai radote greičiausią maršrutą ir tikėjotės, kad nėra greitesnio? O gal išbandėte daugybę variantų ir radote greičiausią?

Kompiuterių programose, skirtose spręsti tokio tipo uždavinius, dažnai naudojamas sisteminis metodas, vadinamas paieška į plotį. Sprendžiant šį uždavinį šiuo metodu būtų atliekami tokie žingsniai:

1. Pažymėkite visus langelius, kuriuos abu draugai gali pasiekti per vieną minutę.
2. Pažymėkite visus langelius, kuriuos galima pasiekti (daugiausia) per vieną minutę iš visų pirmame žingsnyje pažymėtų langelių. Taip pat įsidėmėkite naudojamą transporto priemonę.
3. Pažymėkite visus langelius, kuriuos galima pasiekti per vieną minutę iš 2-ame žingsnyje pažymėtų langelių.

   Kadangi abi iki šiol pažymėtos sritys nepersidengia, draugai negali susitikti po 3 minučių.
4. Dabar pažymėkite visus langelius, kuriuos galima pasiekti per vieną minutę iš 3-ame žingsnyje pažymėtų langelių.

Dabar abi sritys kertasi viename langelyje. Jį per 4 minutes gali pasiekti mergina automobiliu, o vaikinas – dviračiu.

Tikriausiai žinote, kad yra navigacijos sistemų, kurios randa greičiausią maršrutą tarp dviejų taškų, įsitikinus, kad maršrutas eina tinkamais keliais ir takais, o ne per, pavyzdžiui, upes. Ši užduotis panaši į navigacijos uždavinį, tik čia abu draugai turi būti nukreipti į bendrą, iš pradžių nežinomą, tikslą, o ne tik vienas asmuo į fiksuotą paskirties vietą. Kadangi kompiuteris sistemingai atlieka paiešką į plotį, jis randa sprendimus, kurie nėra iš karto akivaizdūs. Kartais aplinkkelis, kuriame yra mažiau šviesoforų, yra greitesnis už trumpiausią kelią nuo pradžios taško iki tikslo. Susisiekimas traukiniu su persėdimais gali būti greitesnis negu tiesioginis susisiekimas autobusu.

Informatikoje žinomi įvairūs metodai, kaip rasti geriausią tokio tipo uždavinio sprendimą. Be ką tik aprašytos paieškos į plotį, yra „šakų ir ribų“ metodas. Atliekant paiešką į plotį, atsižvelgiama į kiekvieną dalinį sprendimą, pasiektą atliekant tam tikrą darbo etapų skaičių. Taikant „šakų ir ribų“ metodą dalinių sprendimų nebesiekiama, jei žinoma, kad jie negali pateikti geresnio sprendimo.

Jei uždavinys būtų labai sudėtingas, net ir sparčiausias pasaulio kompiuteris užtruktų per ilgai, kad rastų visas galimybes geriausiam sprendimui pasiekti. Naudojant navigacijos sistemą dažnai pakanka rasti gerą maršrutą, net jei jis nėra pats geriausias. (Jei galite nuvykti per 78 minutes, tikriausiai neprieštarausite pasirinkti šį variantą, net jei žinote, kad teoriškai galima būtų nuvykti per 77 minutes.)

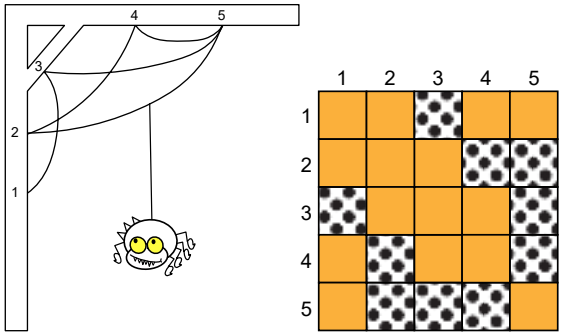
Tokiu atveju gali būti naudojamas vadinamasis godusis algoritmas. Taikant godųjį algoritmą, kiekviename žingsnyje naudojamas tarpinis rezultatas, kuris tuo metu atrodo geriausias. Visada žiūrima tik į dabartinę situaciją, nekreipiant dėmesio į tai, kas gali nutikti toliau. Todėl godusis algoritmas yra daug greitesnis už paiešką į plotį. Jis dažnai pateikia gerą, bet nebūtinai optimalų uždavinio sprendimą.

42. Voro antklodėlės

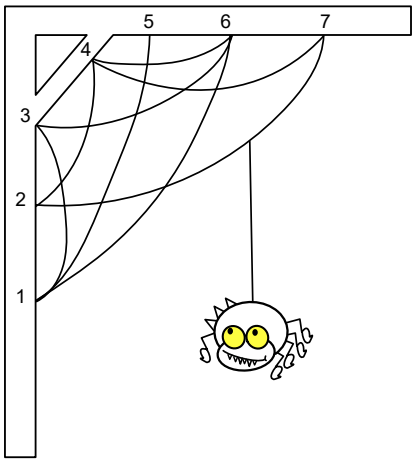
Pamačiusi įdomų voratinklį Vanda jo pagrindu kuria dygsniuotą antklodėlę. Ji vietas, kuriose pritvirtinti voratinklio siūlai, sunumeruoja skaičiais nuo 1 iki N ir iš skiaučių sukuria N x N dydžio antklodėlę, laikydamasi šių taisyklių:

- Kiekvienam voratinklio siūlui, besidriekiančiam nuo vietos X į vietą Y, ji siuva dvi taškuotas skiautes antklodėlėje:
  - Viena taškuota skiautė siuvama eilutės X ir stulpelio Y sankirtoje;
  - Kita taškuota skiautė siuvama eilutės Y ir stulpelio X sankirtoje.
- Likusios skiautės siuvamos vienspalvės.

Pavyzdžiui, šiam voratinkliui gaunama tokia antklodėlė:

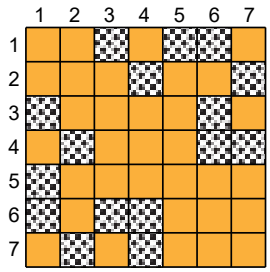


Dabar Vanda mato tokį voratinklį:

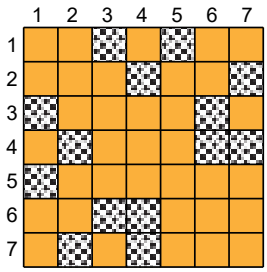


Kaip atrodys pagal šį voratinklį sukurta antklodėlė?

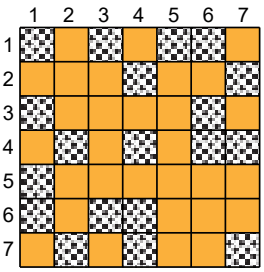
A.



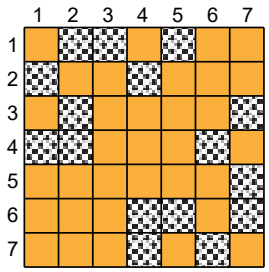
B.



C.



D.



Paaiškinimas

Teisingas atsakymas: A.

Voratinklio siūlas jungia tvirtinimo vietą 1 su vietomis 3, 5 ir 6. Taigi pirmoje eilutėje taškuotos bus 3, 5 ir 6 stulpelio skiautės.

Siūlas jungia tvirtinimo vietą 2 su vietomis 4 ir 7. Taigi antroje eilutėje taškuotos bus 4 ir 7 stulpelio skiautės.

Siūlas jungia tvirtinimo vietą 3 su vietomis 1 ir 6. Todėl trečioje eilutėje taškuotos bus 1 ir 6 stulpelio skiautės.

Siūlas jungia tvirtinimo vietą 4 su vietomis 2, 6 ir 7. Ketvirtoje eilutėje taškuotos bus 2, 6 ir 7 skiautės.

Siūlas jungia tvirtinimo vietą 5 su vieta 1. Penktoje eilutėje taškuota bus tik 1-ojo stulpelio skiautė.

Siūlas jungia tvirtinimo vietą 6 su vietomis 1, 3 ir 4. Šeštoje eilutėje taškuotos bus 1, 3 ir 4 skiautės.

Siūlas jungia tvirtinimo vietą 7 su vietomis 2 ir 4. Septintoje eilutėje taškuotos bus 2 ir 4 skiautės.

Atsakymas B neteisingas, nes nėra taškuotos skiautės 1-os eilutės 6-ame stulpelyje ir 6-os eilutės pirmame stulpelyje.

Atsakymas C neteisingas, nes jame 3 langeliai neteisingai pažymėti kaip taškuoti: 1-os eilutės ir 1-o stulpelio sankirta, 4-os eilutės ir 4-o stulpelio, 7-os eilutės ir 7-o stulpelio sankirta.

Atsakymas D neteisingas, nes visas kilimėlis pasuktas 90 laipsnių kampu.

Tai informatika!

Voratinklis gali būti laikomas grafu. Grafo sąvoka dažnai naudojama informatikoje.

Grafą sudaro viršūnės (voratinklio tvirtinimo vietos) ir briaunos (voratinklio siūlai, jungiantys tvirtinimo vietas). Grafai naudojami pavaizduoti objektams ir ryšiams tarp jų. Pavyzdžiui, grafas gali parodyti ryšius tarp žmonių socialiniuose tinkluose arba skrydžius tarp šalių.

Šioje užduotyje Vandos antklodėlė parodo, kaip dar gali būti vaizduojamas grafas, t. y., naudojant gretimumo matricą. Gretimumo matricos yra naudingos, nes padeda atsakyti į tokius klausimus, kaip „ar konkreti briauna egzistuoja?“, „kiek briaunų yra sujungta su konkrečia viršūne?“.

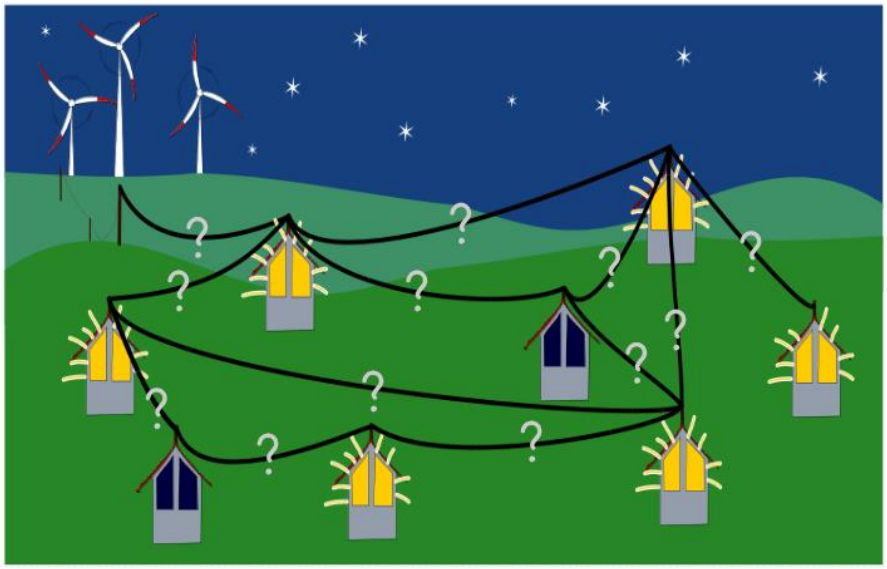
43. Gedimų nustatymas

Bebrų mieste elektrą gamina ant kalvų pastatytos vėjo jėgainės, kurios laidų linijomis sujungtos į bendrą elektros tinklą taip, kaip pavaizduota paveiksle:

Kai kuriose tinklo linijose įvyko gedimai ir elektros srovė nepasiekia dviejų namų. Kituose namuose elektros energija naudojama be trikdžių. Elektros srovė laidais teka nuo bet kurio namo prie bet kurio kito namo bet kuria kryptimi.

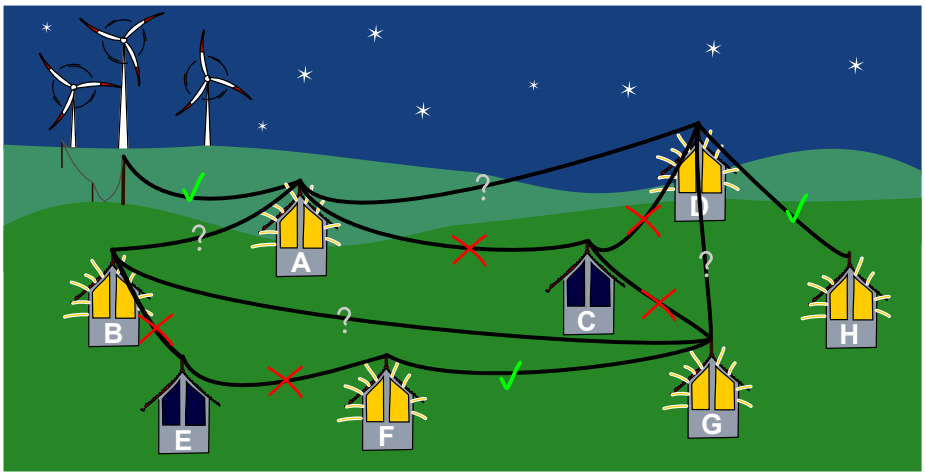
Atsižvelgdami į paveiksle pavaizduotą kiekvieno namo situaciją nustatykite, kokia yra kiekvienos linijos būsena. Jei linijoje yra gedimas, ją pažymėkite ženklu ✗. Jei linijoje gedimo nėra, tada žymėkite ✓. Jei neįmanoma nustatyti, ar linijoje yra gedimas, žymėkite ?.

Paveikslėlyje spustelėjus ant kiekvienos linijos galima parinkti vieną iš trijų būsenų, atitinkančią užduoties sąlygą.



Paaiškinimas

Paveiksle pažymėti namai E ir C neturi elektros, nes visose linijose, jungiančiose šiuos namus tiesiogiai su kitais, yra gedimai.



Į namą H ateina tik viena linija iš D namo, vadinasi, joje gedimo nėra. Linijoje, jungiančioje A namą su vėjo jėgainėmis taip pat gedimo nėra, nes kitaip nei vienas namas neturėtų elektros.

Taip pat F namą su G namu jungianti linija yra veikianti, nes linijoje, jungiančioje F su E yra gedimas.

Apie likusias linijas vienareikšmiškai atsakyti neįmanoma, nes, pavyzdžiui namas B gali gauti elektros iš linijos, jungiančios B namą su G, ir iš linijos, B jungiančios su A. Nors elektros srovė teka abiem kryptimis, tačiau jei bus sugedę abi D–A ir D–G linijos, o D–H linija nepažeista, D namas šia linija elektros negaus.

Bet kuri viena iš A–B–G–D–A tinklo linijų gali būti sugedusi ir A, B, D, G namai vis tiek gaus elektros energiją.

Tai informatika!

Kompiuterių tinkluose, kaip ir elektros skirstymo tinkluose, kai kuriose jungtyse gali būti sutrikimų – jos lėtos, perkrautos arba visiškai sugedusios. Tinklo struktūros papildomos jungtys užtikrina nuolatinį jo pralaidumą gedimų atveju (jei tuo pačiu metu nėra per daug gedimų).

Norėdami pavaizduoti tinklo struktūras, mokslininkai naudoja grafus. Yra daug algoritmų, susijusių su grafais, skirtų efektyviai nustatyti sugedusias tinklo jungtis.

Su sistemos klaidų taisymu mokslininkai dažnai susiduria ne tik kompiuterių tinkluose, bet ir kurdami programinę įrangą. Kad ištaisytume klaidą, tenka nustatyti tikslią jos priežastį, ir šis procesas paprastai atliekamas palaipsniui keliais etapais. Kai kurie programuotojai yra įsitikinę, kad neįmanoma surasti visų programos klaidų.

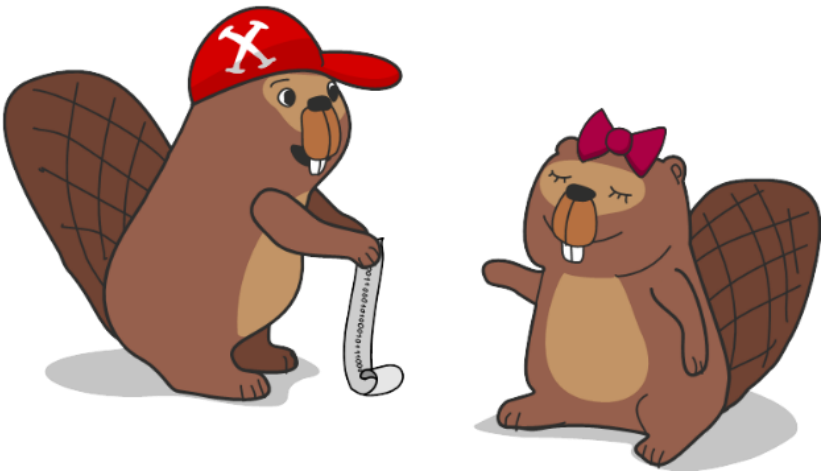
44. Lakoniški pranešimai

Bebras Joris užkodavo kelias raides simboliais 0 ir 1 taip, kaip parodyta lentelėje.

Raidė	L	U	I	M	Z	N
Kodas	1	00	0010	0110	1010	1110

Joris draugei Elzei siunčia žinutę:

1 0 0 1 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 1 1 1 0 0 0



Elzė suprato, kad žinutėje paskutinė užkoduota raidė yra U.

Kokia buvo Jorio žinutė, užrašyta raidėmis?

Paaiškinimas

Atsakymas: LIMUZINU.

Raidė	L	I	M	U	Z	I	N	U
Kodas	1	0010	0110	00	1010	0010	1110	00

Lentelėje Jorio žinutės raidės ir jų kodai surašyti iš eilės.

Kodai yra skirtingo ilgio, tačiau tai nėra priešdėlio kodas (angl. *prefix-free code*), nes U raidės kodas atkartojamas I raidės kode. Todėl norint sėkmingai ir vienareikšmiai perskaityti žinutę, reikia pastebėti, kad pradėjus skaityti iš kairės į dešinę, susiduriama su problema, kaip nustatyti antrąją raidę. Pirmoji, akivaizdu, yra kodą „1“ turinti raidė L, tačiau, kokia antroji raidė, nėra vienareikšmio atsakymo, nes tai gali būti ir U, kurios kodas „00“, ir I, kurios kodas „0010“.

Pradėjus skaityti iš dešinės į kairę, visi kodai tampa vienareikšmiai ir žinutė perskaitoma lengvai. Tokiu atveju tinka priesagos koduotė (angl. *suffix-free*). Visų raidžių, užkoduotų keturiais simboliais kodų pabaigos nesutampa su trumpaisiais kodais.

Užduoties tekste užuomina apie paskutinę užkoduotą raidę U ir yra skirta atkreipti dėmesį į tai, jog žinutę sėkmingai iškoduoti galima ją skaitant nuo galo.

Tai informatika!

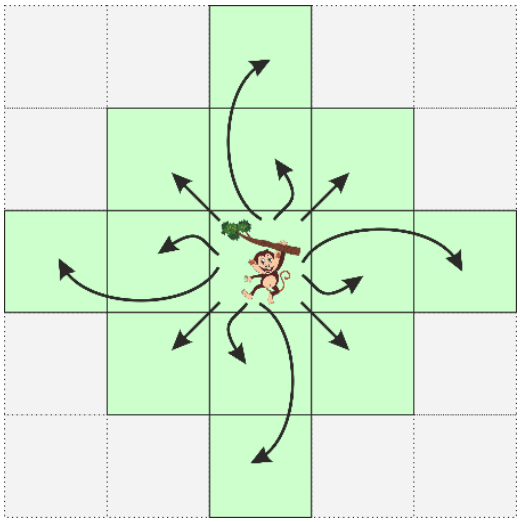
Skaitmeninės technologijos koduoja visą informaciją bitų (vienetų ir nulių) sekomis. Teisingai užkoduota informacija turi būti vienareikšmė. Tai pasiekiama, kai kiekvienas informacijos simbolis (šiame uždavinyje raidė) koduojamas skirtingai. Tačiau kai kurie simboliai pasikartoja žymiai dažniau ir jiems pritaikomas trumpesnis kodas. Toks kodas vadinamas priešdėlio kodu (angl. *prefix-free code*) ir koduojant tokiu būdu vienareikšmiškumas pasiekiamas, naudojantis tokia taisykle: jei trumpas kodas yra *a*, tai ilgas kodas niekada neprasidės *a* kodo seka. Pavyzdžiui, koduotė, kuria yra užkoduoti tik trys simboliai 0, 10 ir 11 yra priešdėlio kodas, nes visa informacija, užkoduota šiais kodais, yra perskaitoma vienareikšmiai.

Turime žinutę 01001101110 ir ją vieninteliu būdu galima išskaidyti taip: 0 10 0 11 0 11 10. Pavyzdžiui, koduotė, sudaryta iš kodų 0, 10, 11 ir 100, nėra priešdėlio kodas, nes žinutę 01001101110 galima perskaityti dviem būdais: 0 10 0 11 0 11 10 ir 0 100 11 0 11 10.

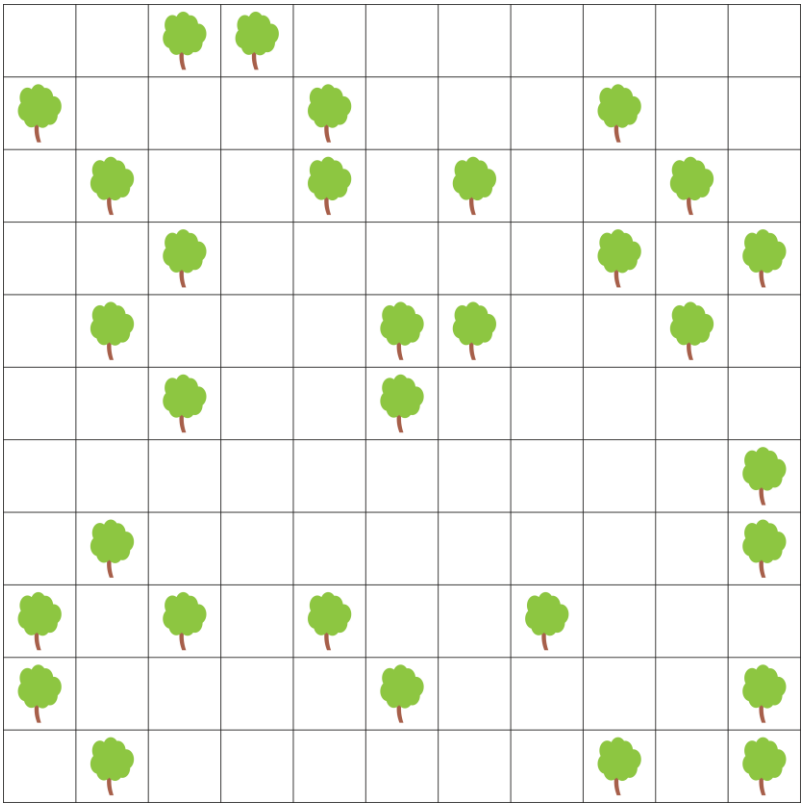
Kodavimui gali būti naudojamas ir priesagos kodas (angl. *suffix-free code*), kai trumpieji kodai nėra ilgųjų kodų pabaigos. Šiame uždavinyje keturių bitų kodai nesibaigia nei vienetu, nei dviem nuliais. Todėl šio uždavinio kodavime naudojamas priesagos kodas.

45. Šoklusis beždžioniukas

Beždžioniukas Jonukas gyvena medžiais apsodintame parke. Parkas padalintas į kvadratėlius. Paveikslėlyje parodyta, kokius šuolius nuo vieno medžio ant kito gali atlikti Jonukas.



Jonuko parkas atrodo taip:

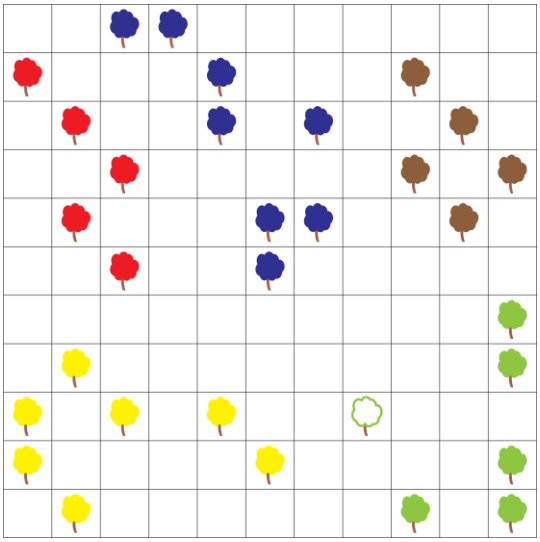


Jonukas nori išsirinkti medžių grupę su daugiausia medžių, tarp kurių galėtų šokinėti nenusileisdamas ant žemės.

Pažymėkite medžių grupę, kurią patartumėte pasirinkti Jonukui. Grupės medžius žymėkite mėlyna spalva, spustelėdami kiekvieną tokį medį.

Paaiškinimas

Paveiksle pavaizduoti 8 mėlyni medžiai, į kuriuos gali nušokti beždžioniukas, nei karto nenusileisdamas ant žemės. Kitos mažesnės medžių grupės nuspalvintos kitomis spalvomis.

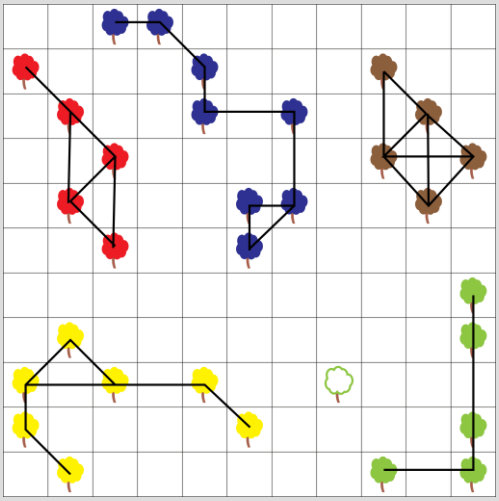


Parke galima išskirti 5 medžių grupes ir vieną vienišą medį, kurie atskirti vieni nuo kitų kvadratėliais, kurių peršokti nenusileisdamas ant žemės beždžioniukas negali.

Kaip tokias grupes surasti? Galima spustelėti atsitiktinai bet kurį medį, jį nuspalvinti. Ta pačia spalva nuspalvinus gretimus medžius, į kuriuos beždžioniukas gali peršokti, ir susidarys viena grupė. Tada nuspalvinus kita spalva bet kurį nepažymėtą medį ir aplink jį ta pačia spalva pažymėjus gretimus medžius, į kuriuos gali nušokti beždžioniukas, susidarys nauja grupė. Taip bus gautos 6 skirtingos medžių grupės.

Tai informatika!

Informatikoje beždžioniuko šokinėjimas vaizduojamas grafu. Medžiai atitinka grafo viršūnes, o šuolis – briauną tarp grafo viršūnių. Paveiksle briaunos vaizduojamos atkarpomis, jungiančiomis du medžius.



Jei randame kelią, kuris sujungia visas briaunas, tai tie medžiai priklauso vienai grupei. Tokios grupės vadinamos sujungtais grafo komponentais. Kiekvieno komponento viršūnės vaizduojamos skirtingomis spalvomis.

46. Truchet'o plytelės

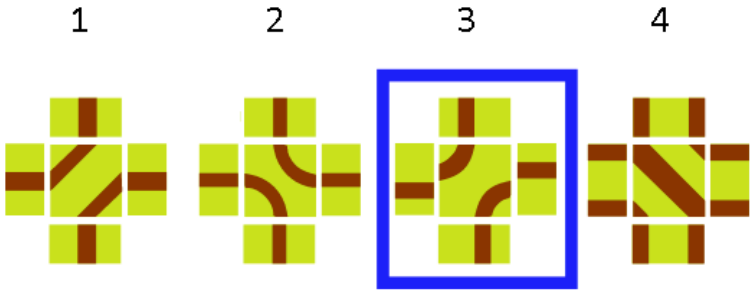
Kiekvienas šių rekursinių ornamentų sukurtas dėliojant tokią pat plytelę. Visos tokios plytelės, kurios naudojamos pavaizduotiems ornamentams kurti, remiasi svarbia bendra savybe.

Kuri iš šių plytelių neturi rekursiniams ornamentams būdingos savybės?

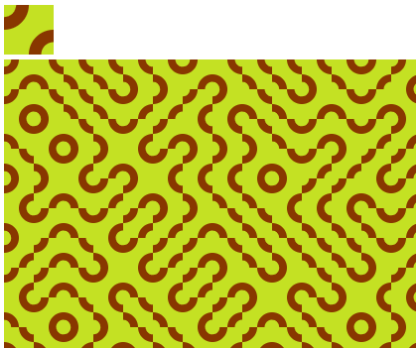
Užuomina:
Atkreipkite dėmesį į plytelių kraštus.

Paaiškinimas

Trečioji plytelė neturi šios savybės.



1, 2 ir 4-os plytelių kraštai baigiasi tais pačiais raštais ir yra vienodi. 3-os plytelės kraštų raštai nesimetriniai: viršutiniai ir apatiniai plytelių kraštai puikiai tinka, bet šoniniai – ne.



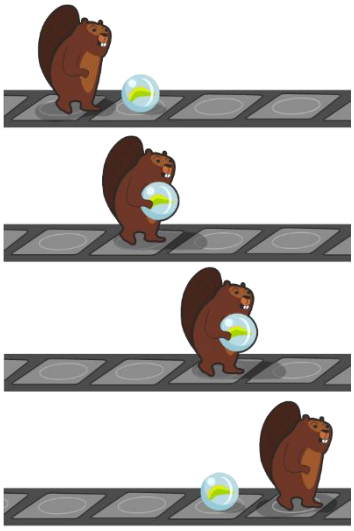
Dėl šios priežasties 3-ios plytelės raštas neatrodo sklandžiai, jis trūkinėjantis.

Parodome, kaip dėliojant 1, 2 ir 4-ą plyteles sukuriami rekursiniai raštai.

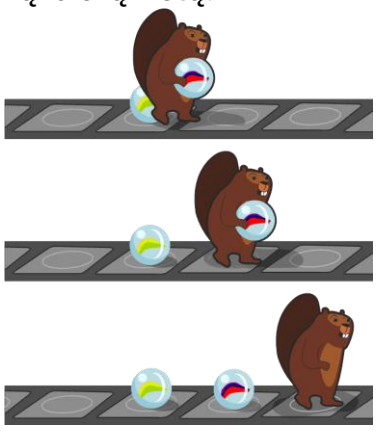
Tai informatika!

Šios plytelės pavadintos prancūzų vienuolio, įvairių išradimų autoriaus Sébastieno Truchet'o vardu, kuris sukūrė šiuos rekursinius plytelių raštus. Plytelės su 4-iais vienodais kraštais sudaro Truchet'o plytelių pogrupį (Truchet'o plytelės nebūtinai turi turėti 4-is tuos pačius kraštus kaip pateikta šio uždavinio pavyzdžiuose). Tai, kad sudėtingus raštus galime sukurti naudodami labai paprastus blokelius (plyteles), žmones žavėjo nuo seno. Truchet'o plyteles nagrinėja ir matematikai, ir informatikai, jos naudojamos kompiuteriniuose žaidimuose, labirintų uždaviniuose, įvairiose dekoracijose.

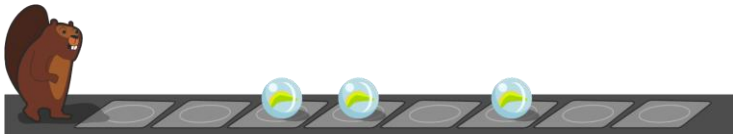
47. Žaidimo taisyklės



Bebras žingsnis po žingsnio juda per juostos langelius. Kai kuriuose juostos langeliuose gali būti marmuro rutuliukų, su kuriais reikia elgtis pagal taisykles. Jei bebras randa rutuliuką, jį paima ir eina toliau, kol pasiekia tuščią langelį, į kurį ir padeda šį rutuliuką.
Bebras vienu metu gali nešti tik vieną rutuliuką, o kiekviename langelyje irgi gali būti tik vienas rutuliukas. Jei bebras turi rutuliuką ir ateina į langelį, kuriame jau yra rutuliukas, jis eina toliau ir turimą rutuliuką įdeda į pirmą laisvą vietą.



Kuriuose langeliuose bus marmuro rutuliukai, kai bebras pereis visą juostą?



A. 

B. 

C. 

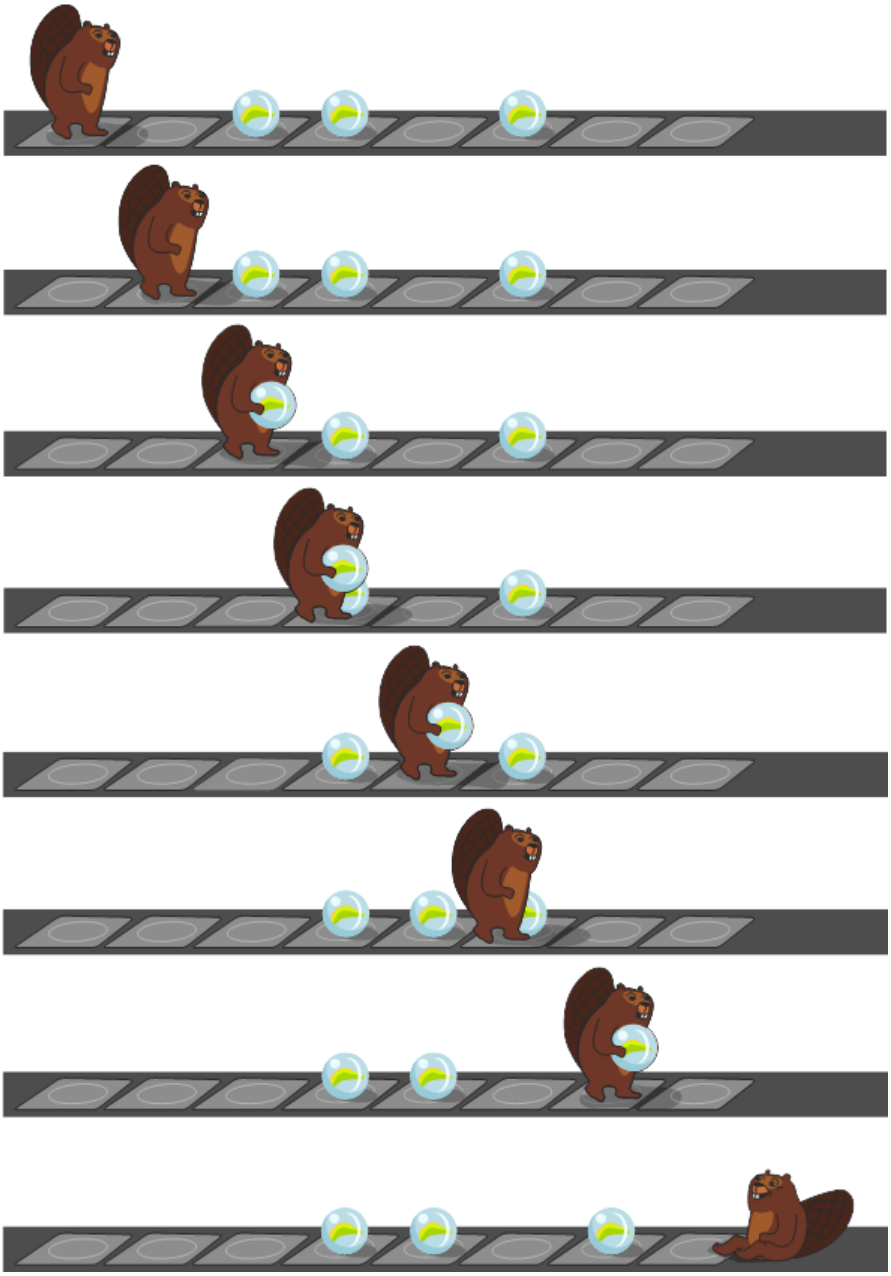
D. 

E. 

Paaiškinimas

Teisingas atsakymas: C.

Tai išplaukia iš bebrų žaidimo taisyklių nuoseklaus vykdymo:



Tai informatika!

Kompiuterių moksle gana paprastos operacijos dažnai duoda įdomių rezultatų. Ši užduotis yra geras to pavyzdys. Pateiktas algoritmas pagrįstas tuo, kad bebras gali priimti dvi skirtingas būsenas (su rutuliuku ar be) ir kad savo kelyje gali rasti du skirtingus langelius (užimtus arba tuščius).

Apskritai šios operacijos rezultatas primena dvejetainį bitų poslinkį, kai dvejetainis skaičius dauginamas iš 2 arba dalijamas iš 2. Kad ir koks paprastas būtų šis pavyzdys, jame slypi keletas esminių Tiuringo mašinos elementų.

Tiuringo mašina yra teorinis labai paprasto kompiuterio modelis. Galima parodyti, kad Tiuringo mašina, nepaisant jos paprastumo, gali vykdyti visus algoritmus, kuriuos gali atlikti bet kuris kompiuteris. Tiuringo mašinos koncepciją pasiūlė anglų matematikas Alanas Tiuringas 1936 metais ir ji daug kartų pasitvirtino (pavyzdžiui, labai praktiška, kad visi loginiai teiginiai, taikomi Tiuringo mašinai, tinka ir visiems klasikiniams kompiuteriams).

Turingo mašiną sudaro:

- Juosta, kurią galima perskaityti ir užrašyti. Mūsų pavyzdyje tai yra langeliai, kuriais juda bebras.
- Ribinė simbolių abėcėlė, kartais sąmoningai apsiribojama 0 ir 1. Mūsų pavyzdyje rutuliukas simbolizuoja 1, o laisva erdvė – 0.
- Skaitymo ir rašymo galvutė, galinti skaityti simbolius iš juostos ir rašyti į ją, ji gali judėti viena ar abiem juostos kryptimis. Mūsų pavyzdyje bebras atlieka skaitymo ir rašymo galvutės vaidmenį.
- Ribotas būsenų rinkinys. Mūsų atveju yra tik būsenos „rutuliuko nešimas“ ir „ėjimas be rutuliuko“.
- Taisyklių rinkinys: kas nutinka, priklausomai nuo būsenos, kai iš juostos nuskaitomas tam tikras simbolis? Galimi veiksmai: būsenos perėjimas, simbolio užrašymas į juostą, skaitymo ir rašymo galvutės perkėlimas į kairę arba į dešinę.

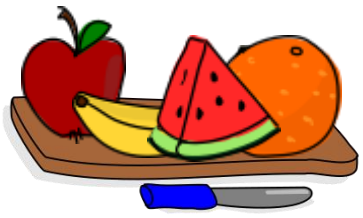
48. Vaisių tvarka

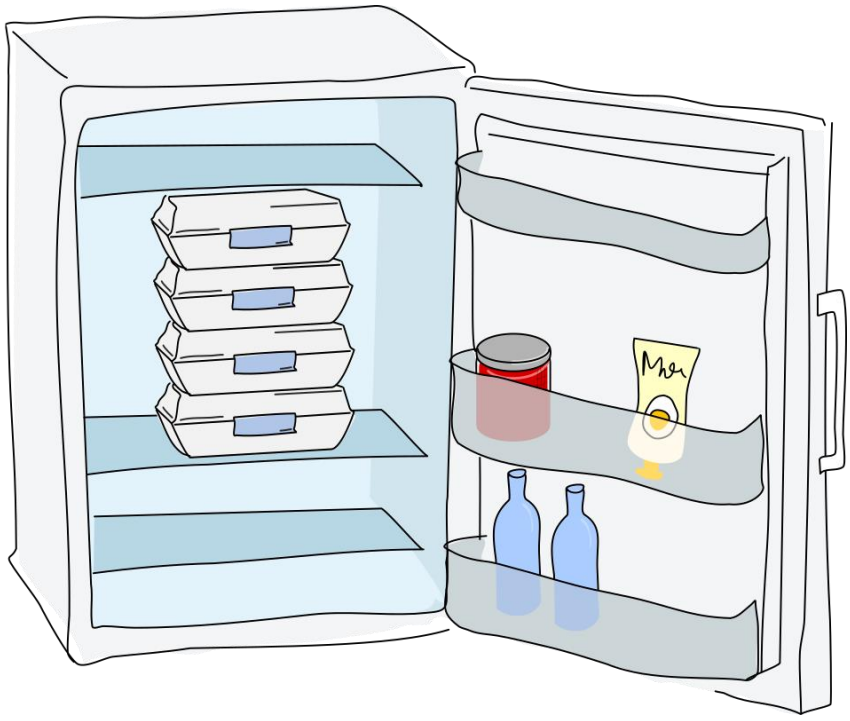
Bebrų šeima – tėtis, mama, Dorė ir Romas – vakare ruošia pusryčių dėžutes, į kurias deda po vieną iš šių vaisių: obuolį, bananą, apelsiną arba arbūzą. Vaisiai negali kartotis. Paruoštas dėžutes deda vieną ant kitos į šaldytuvą. Rytais bebrai būna dar mieguisti ir išeidami iš namų ima po vieną dėžutę nuo viršaus.

Tiksliai nėra žinoma, kokia tvarka bebrai išeina iš namų, bet tikrai aišku, kad tėtis išeina paskutinis, o mama visada anksčiau už Dorę. Lentelėje parodyta, kokie vaisiai yra tinkami atskiriems šeimos nariams:

				
Tėtis			✓	
Mama	✓		✓	✓
Dorė	✓	✓	✓	
Romas	✓	✓		✓

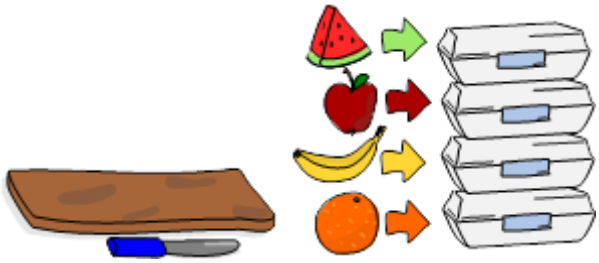
Nuvilkite vaisius į dėžutes taip, kad išeidamas kiekvienas šeimos narys gautų jam tinkamą vaisių.





Paaiškinimas

Yra tik viena galimybė taip sudėti vaisius į dėžutes, kad visi šeimos nariai gautų jiems tinkamus vaisius:



Kaip tai sužinoti? Tėčiui tinka tik apelsinai ir jis išeina paskutinis. Taigi apelsinas turi būti apatinėje dėžutėje. Romui apribojimų nėra. Mama būtinai išeina prieš Dorę, todėl yra galimi trys išėjimo tvarkos variantai:

1	Mama	Mama	Romas
2	Dorė	Romas	Mama
3	Romas	Dorė	Dorė
4	Tėtis	Tėtis	Tėtis

Pagal šią schemą antra gali išeiti mama, Dorė ir Romas, kuriems visiems tinka obuolys. Tuo tarpu mamai ir Romui tiks arbūzas, o Dorei ir Romui – bananas. Todėl pirmiausia reikia dėti dėžutę su apelsinu, ant jos bananą, obuolį ir viršuje arbūzą.

Tai informatika!

Teisinga tvarka yra labai svarbi daugelyje kompiuterių mokslo sričių: daugeliui skaičiavimų reikia tarpinių rezultatų, kurie pirmiausia turi būti nustatyti prieš gaunant galutinį rezultatą. Jei skaičiavimo veiksmai atliekami skirtingais kompiuteriais, nesant kruopštaus planavimo gali atsirasti vadinamosios „aklavietės“. Tai situacijos, kai du ar daugiau procesų laukia vienas kito ir tokiu būdu programa niekada nesibaigia.

Neteisinga tvarka dažnai sukelia įvairių klaidų (bebrų maisto atveju gali būti skrandžio sutrikimų 😊). Pavyzdžiui, jei reikia apskaičiuoti formulę Z = (A + B) × (A - B), ją galima suskirstyti į:

Įvesk A

Įvesk B

Skaičiuok X = A + B

Skaičiuok Y = A – B

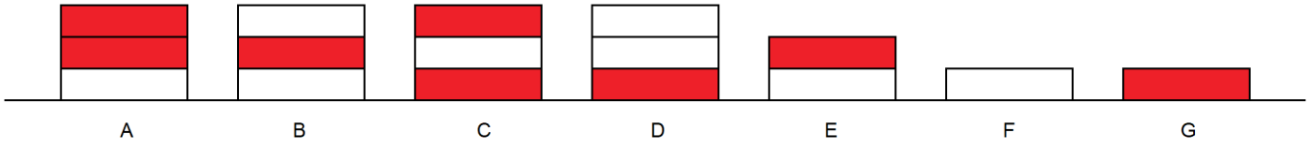
Skaičiuok Z = X × Y

Matome kad įvedimas būtinai turi būti atliekamas pradžioje, tuo tarpu X ir Y skaičiavimo tvarka gali būti sukeista.

Dėl tokių ribojimų daugelyje įprastų programavimo kalbų instrukcijų tvarka yra svarbi. Net ir loginėse, deklaratyviose programavimo kalbose, pvz., „Prolog“, faktų ar taisyklių pateikimo tvarka kartais yra svarbi.

49. Žetonų bokštai

Steponas iš baltų ir raudonų žetonų sudėjo ant stalo septynis bokštelius, pavaizduotus paveikslėlyje.



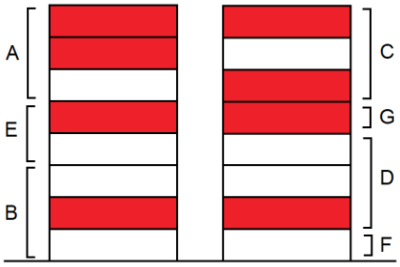
Jis nori sudėti šiuos bokštelius vieną ant kito (nedalindamas) taip, kad susidarytų du visiškai vienodi bokštai. Abiejų bokštų aukštis turi būti vienodas (po aštuonis žetonus) ir žetonai bokštuose turi būti sudėlioti ta pačia spalvų seka (iš apačios į viršų).

Kuris iš šių keturių sprendimų NĖRA tinkamas, jei bokštas (x, y, z, ...) konstruojamas dedant *y* bokštelį ant *x* bokštelio, *z* ant *y* ir t. t.:

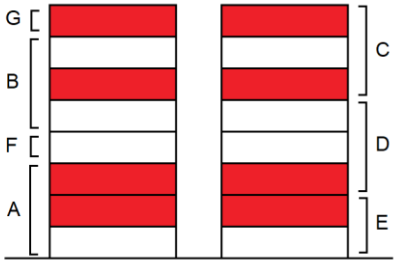
1. (A, F, B, G) ir (E, D, C)
2. (B, A, E) ir (F, D, G, C)
3. (B, E, A) ir (F, D, G, C)
4. (B, E, A) ir (F, D, C, G)

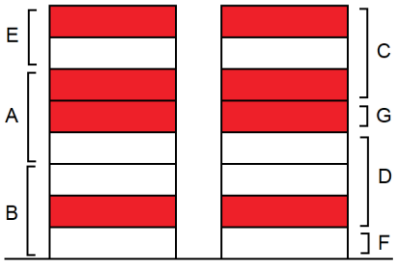
Paaiškinimas

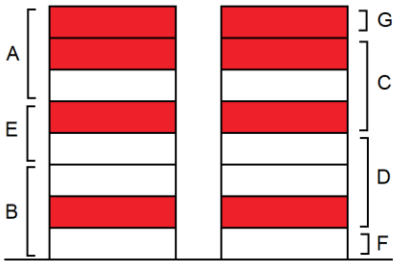
Atsakymas: 3. Netinkamas yra 3-as atsakymo variantas (B, E, A) ir (F, D, G, C), pavaizduotas paveiksle.



Visi kiti pateikti variantai yra galimi teisingi šio uždavinio sprendimai:







Tai informatika!

Įsivaizduokite, kad turite maišelį su daugybe monetų ir norite jas padalyti į dvi vienodos vertės dalis. Jei visos monetos būtų vienodos vertės ir lygios, tada problema būtų lengvai išspręsta: užtektų jas padalyti į dvi krūveles, turinčias vienodą skaičių monetų; bet jei monetų vertės skirtingos, tuomet užduotis tampa šiek tiek sunkesnė.

Matematikos terminais kalbant dalinimo uždavinys reikalauja nuspręsti, ar teigiamų sveikųjų skaičių multiaibė (t. y., aibė su daug galimų reikšmių kiekvienam elementui) gali būti padalyta į du multipoaibius taip, kad pirmojo poaibio skaičių suma – multiaibė – yra lygi skaičių antrajame multipoaibyje sumai. Šis matematinis uždavinys nėra baigtas spręsti (t. y., praktiškai nėra žinomi jo sprendimo variantai, kurie bet kokiu atveju būtų veiksmingi); tačiau jį galima išspręsti naudojant pseudodaugianario laiko algoritmą, paremtą dinaminiu programavimu.

Pateiktoji užduotis nėra paprastesnė už dalinimo uždavinio sprendimą: iš tikrųjų nepakanka, kad dviejų gautų bokštų tik aukštis būtų vienodas, tačiau ir žetonų spalvų seka taip pat turi būti ta pati. Tam gali būti panaudotas pilno perrinkimo metodas ir išsamios paieškos algoritmas.

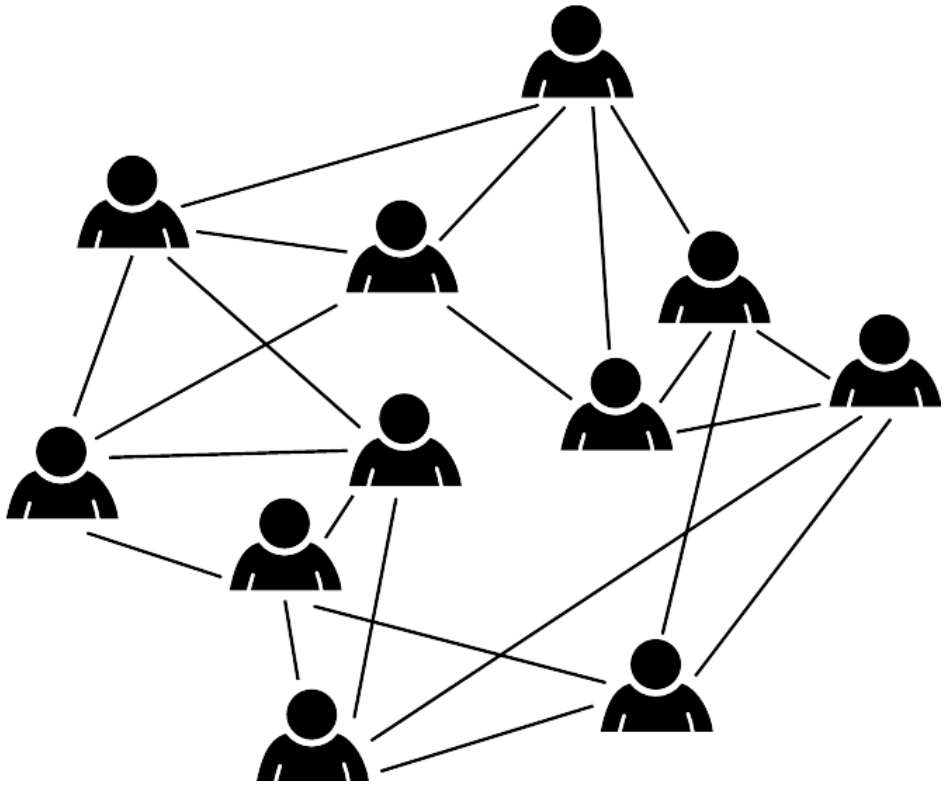
Šiai užduočiai išspręsti iš pradžių surašykime visus galimus septynių bokštų dėliojimo poaibius, kurių kiekvieno suma yra aštuoni žetonai: {A, B, E}, {A, B, F, G}, {A, C, E}, {A, C, F, G}, {A, D, E}, {A, D, F, G}; čia galime sustoti, nes tęsdami rastume šių jau išvardytų rinkinių papildymus. Be to, galime pašalinti aibes {A, C, E} ir {A, C, F, G}, nes joms priklausančiuose bokštuose yra po penkis raudonus žetonus. Todėl lieka tik keturi variantai; kiekvienam iš jų ir atitinkamam papildymui dabar reikia išbandyti visus įmanomus sudėliojimo būdus, kad būtų galima patikrinti, ar kuris nors iš jų sukuria du identiškus bokštus... Mūsų atveju aibė {A, B, E} turi du sprendimo variantus, o aibė {A, B, F, G} lemia tik vieną sprendimą. Tuo tarpu paskutinės dvi aibės neturi jokio sprendimo varianto. Tuomet darosi aišku, kad tokiam sprendimo variantui reikalingas laikas labai greitai auga, jei pradinis bokštų skaičius (mūsų atveju tik septyni) didėja.

Kombinatoriniai (optimizavimo) uždaviniai iškyla daugelyje skirtingų sričių tiek realioje praktikoje, tiek šiuolaikinėse informatikos srityse (pvz., dirbtinio intelekto, mašininio mokymosi srityse).

50. Audito komitetas

Bebrėnų miesto tarybos nariai yra susiję su kitais tarybos nariais kaip kolegos, šeimos nariai, tos pačios politinės partijos nariai ar verslo partneriai.

Tarybą sudaro 11 narių. Jei du tarybos nariai yra kažkaip susiję, jie diagramoje sujungiami linija.



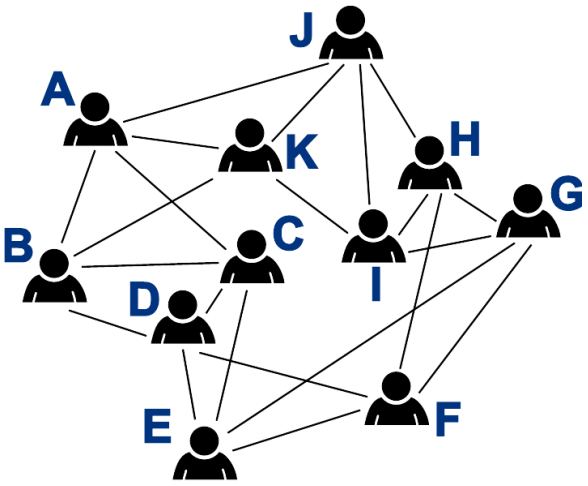
Tarybos audito komitetą sudaro tik tie tarybos nariai, kurie nėra susiję nei su vienu kitu komiteto nariu.

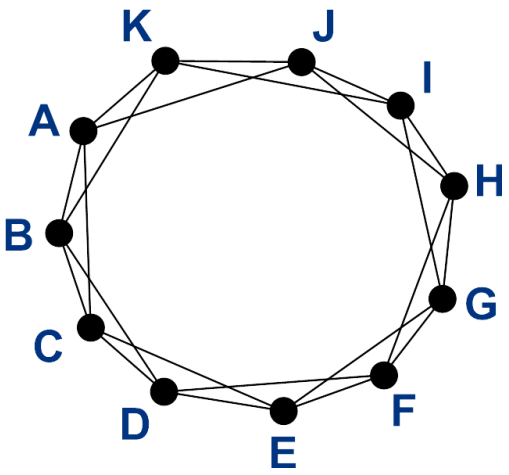
Kiek daugiausiai narių gali būti audito komitete?

Paaiškinimas

Atsakymas: 3.

Patogumo dėlei sužymėkime asmenis raidėmis. Galima pastebėti, kad kiekvienas asmuo yra susijęs su keturiais kitais asmenimis. Dar geriau tai matosi, jei grafą pertvarkysime taip, kaip parodyta dešinėje.





Pavyzdžiui, taškas C yra susijęs tik su A, B, D ir E.

Galime ieškoti kandidatų į audito komitetą nuo bet kurios vietos, nes diagrama neturi ypatingų taškų.

Jei mes pradėsime nuo A, artimiausias pagal laikrodžio rodyklę su A nesusijęs taškas yra D. Kitas su D nesusijęs taškas yra G, o po jo – J. Tačiau J yra susijęs su A ir negali būti audito komitete. Tik 3 nariai A, D ir G yra nepriklausomi.

Tai informatika!

Ryšiai tarp tarybos narių modeliuojami grafu. Grafą sudaro viršūnės (tarybos nariai, taškai) ir briaunos (juos jungiančios linijos). Abi diagramos vaizduoja tą patį grafą.

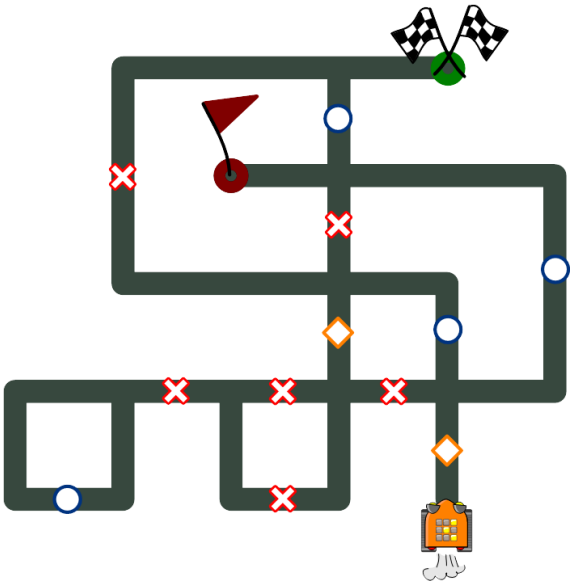
Grafas yra abstrakti struktūra, ji naudinga kalbant apie jungtis: grafas pabrėžia svarbias savybes (kas su kuo susijęs) ir padeda atsiriboti nuo mažiau svarbių (susijusių asmenų savybės, tarpusavio ryšių pobūdis).

Kompiuteriai gali apdoroti grafus labai efektyviai, todėl informatikai turi suprasti grafus, jų rūšis ir savybes.

Naudojant grafų terminiją, ši užduotis reikalauja rasti nepriklausomą aibę tarp grafo viršūnių, t. y., poaibį grafo viršūnių, kurių nesieja briauna. Konkrečiai čia mes ieškome grafo nepriklausomumo lygmens, t. y., didžiausios galimos nepriklausomos aibės dydžio.

51. Simbolių skaitymo robotas

Robotas pradeda nuo paveiksle parodytos pozicijos ir juda nubraižytomis linijomis. Kaip robotui judėti artimiausioje sankirtoje nurodoma simboliais , ir .



Robotui draudžiama nukeliauti iki vėliavėlės .

Kiekvieno simbolio veiksmai skirtingi ir robotui pasiekus sankirtą jie gali reikšti:

- pasisukti kairėn

arba

- pasisukti dešinėn

arba

- judėti tiesiai

Deja, nežinoma, kokius veiksmus reiškia kiekvienas simbolis.

Simbolio reikšmė nesikeičia keičiantis roboto judėjimo krypčiai.

Šiuose dviejuose paveikslėliuose rodyklė rodo, kaip robotas pasisuktų, jei trikampio simbolis reikštų „pasisukti kairėn“:

Padėk robotui teisingai pasirinkti simbolių veiksmus ir pasiekti finišą .

A) = pasisukti dešinėn, = judėti tiesiai, = pasisukti kairėn

B) = pasisukti kairėn, = judėti tiesiai, = pasisukti dešinėn

C) = pasisukti dešinėn, = pasisukti kairėn, = judėti tiesiai

D) = judėti tiesiai, = pasisukti dešinėn, = pasisukti kairėn

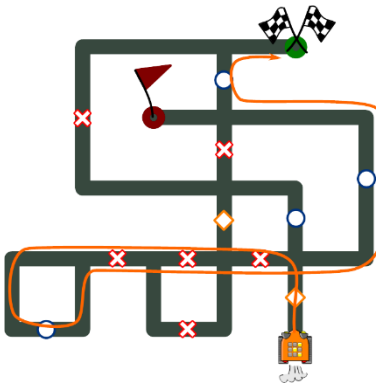
E) = pasisukti kairėn, = pasisukti dešinėn, = judėti tiesiai

F) = judėti tiesiai, = pasisukti kairėn, = pasisukti dešinėn

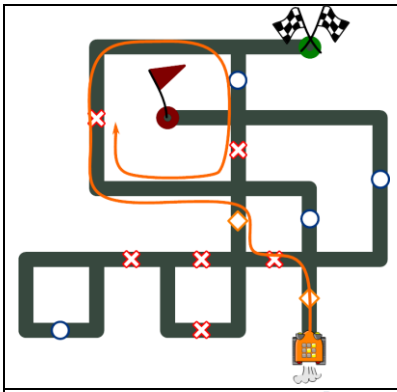
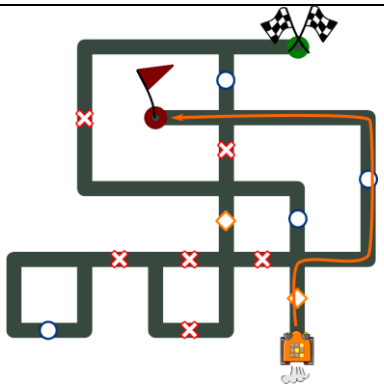
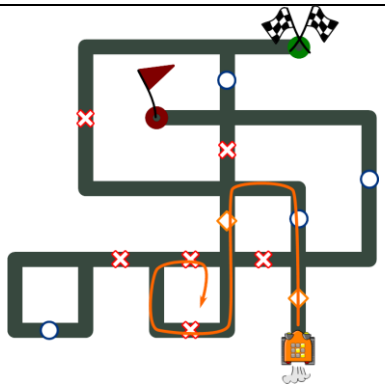
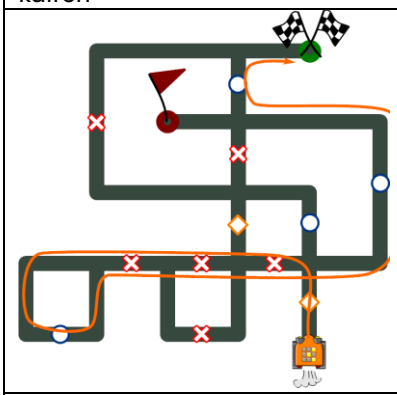
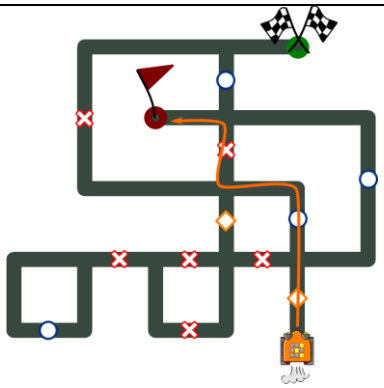
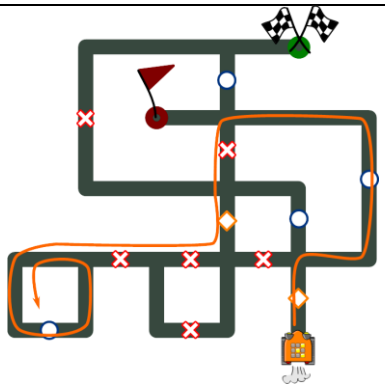
Paaiškinimas

Teisingas atsakymas: D.

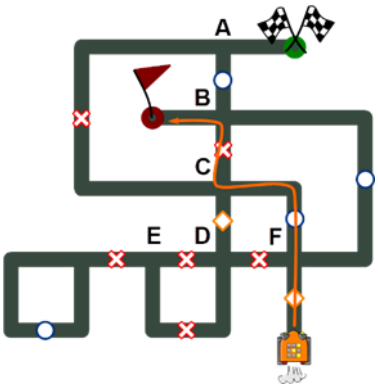
(✕ = judėti tiesiai, ○ = pasisukti dešinėn, ◇ = pasisukti kairėn)



Žinant, kad įmanomos 6 skirtingos galimybės, galima naudoti paprasčiausią uždavinio sprendimo strategiją, kai šios galimybės peržiūrimos ir surandamas vienintelis sprendimas, robotą nuvedantis prie finišo.

		
✕ = pasisukti dešinėn, ○ = judėti tiesiai, ◇ = pasisukti kairėn	✕ = pasisukti kairėn, ○ = judėti tiesiai, ◇ = pasisukti dešinėn	✕ = pasisukti dešinėn, ○ = pasisukti kairėn, ◇ = judėti tiesiai
		
✕ = judėti tiesiai, ○ = pasisukti dešinėn, ◇ = pasisukti kairėn	✕ = pasisukti kairėn, ○ = pasisukti dešinėn, ◇ = judėti tiesiai	✕ = judėti tiesiai, ○ = pasisukti kairėn, ◇ = pasisukti dešinėn

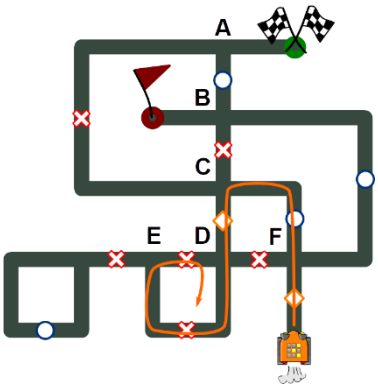
Kita strategija – panagrinėti vieną galimą roboto kelią ir suteikti simboliams atitinkamas reikšmes. Jei robotas pasiekia raudoną vėliavėlę arba patenka į begalinį ciklą, tada reikia pasirinkti kitą kelią arba pakeisti simbolių reikšmes.



Kai robotas pasiekia sankirtą F, jis turi tris galimybes: pasisukti kairėn, pasisukti dešinėn, judėti tiesiai.

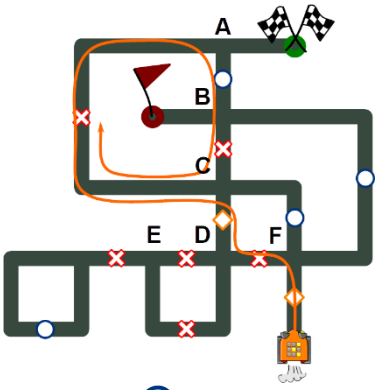
Pirmas spėjimas. Tarkime, kad robotas juda tiesiai. Tuomet simbolis ◇ reiškia veiksmą „judėti tiesiai“. Robotas pasiekia sankirtą C ir ○ gali turėti dvi reikšmes: pasisukti kairėn arba pasisukti dešinėn.

Pirmo spėjimo pirmas variantas. Tarkime, kad ○ reiškia „pasisukti dešinėn“, o ✕ reiškia „pasisukti kairėn“. Tokiu atveju robotas sankirtoje B pasuks kairėn ir nukeliaus prie draudžiamos vėliavėlės ⚑.



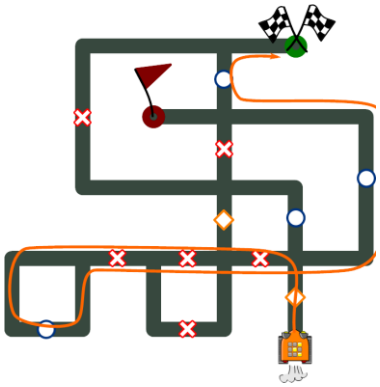
Pirmo spėjimo antras variantas. Tarkime, kad ○ reiškia „pasisukti kairėn“, o ✕ reiškia „pasisukti dešinėn“. Tokiu atveju robotas sankirtoje D keliaus tiesiai ◇, pasieks sankirtą E, pasisuks dešinėn, vėl pasieks D, pasisuks dešinėn ir taip toliau (pateks į amžinąjį ciklą).

Antras spėjimas. Tarkime, kad robotas pasisuka kairėn. Tuomet ◇ simbolis reiškia veiksmą „pasisukti kairėn“.



Antro spėjimo pirmas variantas. Kai robotas pasiekia sankirtą D, vėl yra trys keliai. Tačiau pasisukti kairėn robotas negali (nes buvo kitas simbolis). Pasirenkame, kad ✕ reiškia „pasisukti dešinėn“, o ○ reiškia „judėti tiesiai“. Tuomet robotas sankirtoje C pasisuka kairėn, sankirtoje B juda tiesiai, sankirtoje C pasisuka dešinėn ir patenka į amžinąjį ciklą.

Antro spėjimo antras variantas. Pasirenkame, kad ✕ reiškia „judėti tiesiai“, o ○ reiškia „pasisukti dešinėn“. Dabar robotas pasiekia finišą.



Tai informatika!

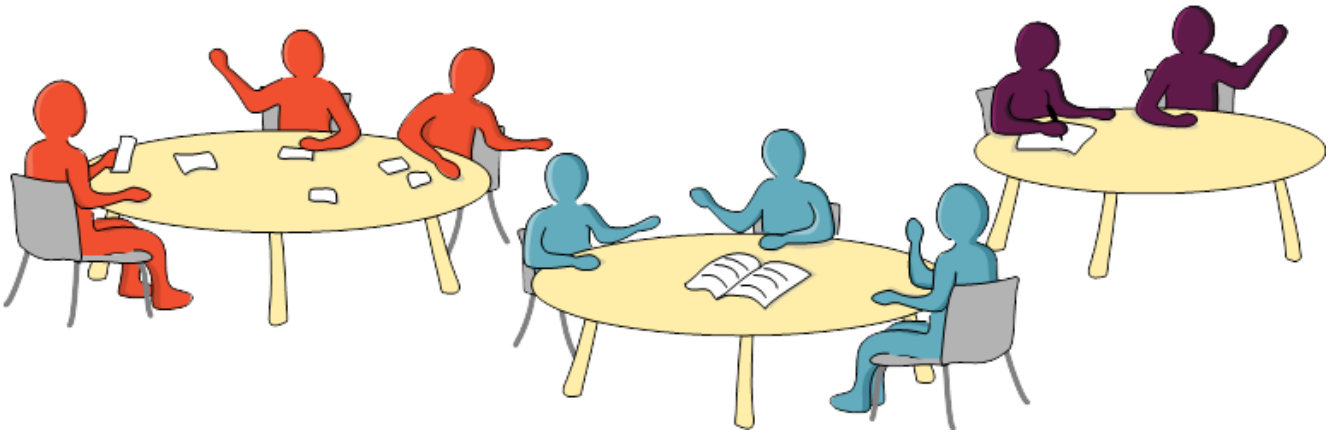
Pirmoji strategija vadinama perrinkimo algoritmu. Nagrinėjamos visos galimybės, kurios galėtų atvesti link norimo rezultato.

Antroji strategija – tai grįžimų (angl. *backtracking*) algoritmas, kai pasirenkami sprendimų variantai, jiems esant nesėkmingiems išbandomi nauji variantai. Tai atveda prie greitesnio atsakymo, nei naudojant pilnąjį perrinkimo metodą.

Šiame uždavinyje kintamųjų mažai, todėl sunku įžvelgti antrosios strategijos efektyvumą. Sprendžiant sudėtingesnius atvejus, grįžimų metodas veda prie greitesnio ir elegantiškesnio sprendimo.

52. Komandų būrimas

Aštuoni mokiniai dažniausiai dalijami į tris grupes užduotims atlikti.



Grafo viršūnės vaizduoja mokinius, o viršūnių spalvos – mokinio komandą. Briaunos sujungia mokinius, kurie nenori mokytis drauge. Deja, vienas stalas, esantis dešinėje, sulūžo ir šios violetinės spalvos komandos abu mokinius tenka įtikinti, kad jie prisijungtų prie skirtingų komandų. Taip mokiniai perskirstomi į dvi komandas po keturis narius.

Spustelėkite vieną grafo briauną, kurią pašalinus mokinius būtų galima suskirstyti į dvi naujas komandas.

Paaiškinimas

Dviejų mokinių įtikinimas mokytis su kitomis komandomis reiškia vienos briaunos panaikinimą. Todėl visos viršūnės gali būti arba raudonos, arba mėlynos. Tos pačios spalvos viršūnės negali turėti bendros briaunos. Lieka vienintelis įmanomas būdas, kai pašalinama oranžine spalva pažymėta briauna.

Pašalinus šią briauną, pasikeičia ir dešinėje pusėje esančių viršūnių spalvos.

Įsitikinkime, kad tai vienintelis teisingas sprendimas. Panagrinėkime iš oranžinės spalvos briaunų sudarytą trikampį.

Jei bent vieną likusią briauną už šio trikampio panaikinsime, vėl reikės trijų spalvų šio trikampio viršūnėms.

Panagrinėkime penkiakampį iš briaunų, nuspalvintų oranžine spalva.

Jei bent vieną likusią briauną už šio penkiakampio panaikinsime, vėl reikės trijų spalvų šio penkiakampio viršūnėms. Lieka vienintelis sprendimas – panaikinti briauną, kuri išardytų turinčius nelyginį skaičių briaunų trikampį bei penkiakampį.

Tai informatika!

Kai kurie uždaviniai yra sprendžiami, kai grafo viršūnėms priskiriamos skirtingos spalvos taip, kad gretimos viršūnės būtų skirtingų spalvų. Šiame uždavinyje viršūnių spalvų skaičius rodo, į kiek grupių gali būti paskirstyti mokiniai. Jei panaikinus vieną briauną viršūnės spalvojamos mažesniu spalvų skaičiumi, tokia briauna vadinama kritine. Šiame uždavinyje tai reiškia, kad du mokiniai pakeičia nuomonę ir sutinka dirbti su kitais mokiniais, todėl gaunamas mažesnis grupių skaičius.

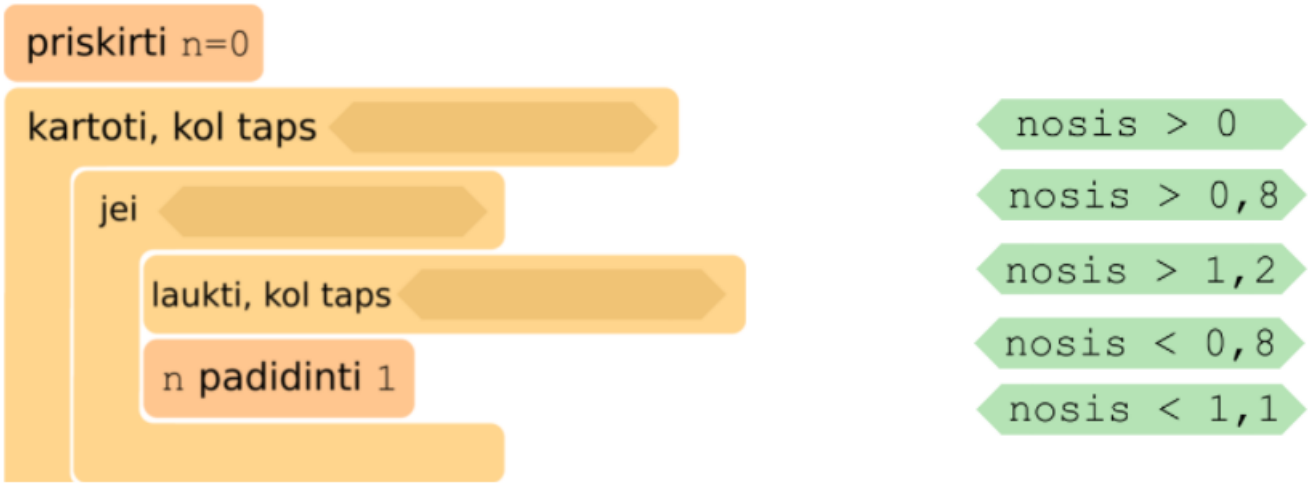
53. Linktelėjimų skaičiavimas

Bilietus parduodantis automatas naudojasi pirkėjo atvaizdu, gaunamu realiu laiku. Norėdamas pirkti *n* bilietų, pirkėjas stovi prieš bilietų automatą ir žvelgdamas į jo kamerą (skaitmeninę akį) *n* kartų linkteli ir po to pakelia galvą. Bilietų automatas analizuoja pirkėjo atvaizdą ir nustato jo nosies nugarėlės ilgį, kurį priskiria kintamajam *nosis*.

Jei kintamojo *nosis* reikšmė yra tarp 0,9 ir 1,1, tai galva yra nepasvirusi ir nepakelta (normali būsena).	nosis =1
Kai pirkėjas linkteli ir galva pasvyra žemyn, tai kintamojo *nosis* reikšmė yra daugiau už 1,2, nes nosies nugarėlės ilgis padidėja.	nosis =1.3
Kai galva pakelta, kintamojo nosis reikšmė tampa mažesne už 0,8.	nosis =0.7

Bilietų automato programa pradeda veikti, kai pirkėjas stovi prie automato ir jo galva yra normalioje pozicijoje.

Užbaikite kurti bilietų pardavimo automato programą nutempdami į jos tuščius tarpus tinkamus sąlygos blokus.



Paaiškinimas

Programa naudoja du kintamuosius *n* (bilietų skaičių atitinkantis linktelėjimų skaičius) ir *nosis* (registruojamos nosies nugarėlės ilgis), kurie atnaujinami automatiškai.

Kol galva pakelta ir nosies nugarėlės ilgis yra mažesnis už 0,8, cikle kartojamos 3 komandos, kurios valdo kintamojo *n* reikšmę. Iš pradžių sistema laukia, kol galva nusileis (nosis > 1,2), o po to – kol pakils (nosis < 1,1). Įvykus vienam linktelėjimui, kintamasis *n* padidėja vienetu.

Kai pirkėjas pakelia galvą (nosies nugarėlės ilgis tampa mažesnis nei 0,8), ciklas užbaigiamas, kintamojo *n* reikšmė atitinka linktelėjimų skaičių ir automatas išduoda pirkėjui *n* bilietų.

Programoje naudojamos nelygybės, nes, kaip ir realiame gyvenime, sunku gauti tikslią nosies reikšmę nosis = 1, kuri reikštų kiekvieno linktelėjimo užbaigimą. Atitinkamai pradinės reikšmės turi būti pakankamai ženklios, kad jos parodytų tikrus pirkėjo ketinimus, o ne nežymų nesąmoningą galvos pasukimą ar mažą linktelėjimą.

Tai informatika!

Skaitmeninė akis (angl. *computer vision*, CV) vartotojui leidžia susikalbėti su automatu gestais.

Pavyzdžiui, elektroninių knygų skaitytuvas su išmania skaitmeninės akies valdymo sistema leidžia negalią turintiems ir negalintiems naudotis rankomis žmonėms vartyti puslapius galvos judesiais. Programavimo kalboms yra specialios bibliotekos, pvz., „OpenCV“, palaikančios skaitmeninės akies funkciją. Šiose bibliotekose yra specialių komandų, kurios leidžia kameros vaizde aptikti įvairias veido dalis ir jas analizuoti.

54. Rąstų rikiavimas

Skirtingo dydžio rąstai plaukia upe. Bebras Hamidas turi juos išrikiuoti pagal ilgį. Todėl jis eina palei upės krantą, sustodamas tik tarp dviejų gretimų rastų. Sustojęs jis palygina šiuos rąstus ir, jei reikia, sukeičia juos vietomis.

Hamidas žino, kad nepaisant to, kokia yra pradinė rąstų tvarka, juos gali išrikiuoti šiuo būdu:

Pradėti eiti iš kairės nuo tarpo tarp pirmojo ir antrojo rąsto.

Kartoti, kol dešinėje yra rąstų:

- *Jei kairysis rąstas yra trumpesnis už dešinįjį: paeiti į dešinę per vieną rąstą.*
- *Jei dešinysis rąstas yra trumpesnis už kairįjį:*
  - *sukeisti šiuos rąstus;*
  - *jei nestovi pradinėje pozicijoje: paeiti į kairę per vieną rąstą.*

Pavyzdyje parodyta, kaip Hamidas šiuo būdu išrikiuoja 6 rąstus. Jam prireikė 17 žingsnių.

Žinoma, žingsnių kiekis, reikalingas išrikiuoti rąstus, priklauso nuo pirminės rąstų tvarkos. Tačiau šešiems rąstams išrikiuoti visada reikės ne daugiau nei 30 žingsnių.

Dabar Hamidas turi išrikiuoti pagal ilgį 60 rąstų.

Kiek žingsnių gali prireikti Hamidui šiems 60 rąstų išrikiuoti?
(Rąstų sukeitimas nėra laikomas žingsniu.)

a) [0...30] Mažiausiai – 0, daugiausiai – 30
b) [6...70] Mažiausiai – 6, daugiausiai – 70
c) [59...300] Mažiausiai – 59, daugiausiai – 300
d) [59...3600] Mažiausiai – 59, daugiausiai – 3600

Paaiškinimas

Variantas a) neteisingas.
Net ir geriausiu atveju, kai rąstai iš pradžių jau yra surikiuoti teisinga tvarka, Hamidui teks pereiti 59 rąstus į dešinę, kad užbaigtų darbą.

Variantai b) ir c) taip pat neteisingi.
Kad tą įrodytume, nagrinėkime atvejį, kuomet rąstai iš pradžių surikiuoti priešinga (mažėjančia) tvarka. Todėl šiuo atveju, kai Hamidas pasiekia k-tąjį rąstą, jį perkelia į pirmą (kairiausią) poziciją. Taigi, k-tajam rąstui perkelti į teisingą poziciją Hamidui reikia paeiti k-2 kartų į dešinę, kad jį pasiektų, ir k-2 kartų paeiti į kairę, kad perkeltų jį į pirmą poziciją. Galiausiai visus surikiavus Hamidas turės vėl pereiti surikiuotus rąstus, taigi dar 59 žingsniai. Todėl gauname: $2\cdot(1+2+3+...+58)+59=592=3481$, kas yra daugiau, nei b) ir c) variantuose parašyta.

Variantas d) teisingas.
Tam turime įrodyti, kad blogiausias atvejis yra priešinga tvarka surikiuoti rąstai. Kai Hamidas pasieka k-tąjį rastą, visi rąstai kairėje jau yra teisingai išrikiuoti. Jam tereikia perkelti šį naują rąstą į teisingą vietą tarp kitų ir pereiti prie (k+1)-ojo rąsto. Todėl anksčiau aptartas atvejis, kai pradinė rąstų eilė yra surikiuota priešinga tvarka, yra pats blogiausias, nes ties kiekvienu rąstu Hamidas turi nueiti maksimalų atstumą iki pradžios. Todėl Hamidui reikės daugiausiai 3481 žingsnių.

Tai informatika!

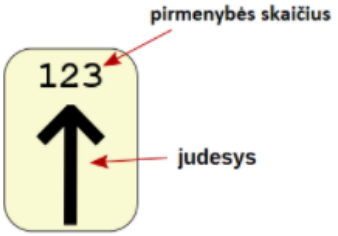
Kaip efektyviai surikiuoti duomenis kokia nors tvarka yra klasikinis informatikos uždavinys. Efektyvus rikiavimas taip pat svarbus optimizuojant ir kitų algoritmų našumą, pavyzdžiui, paieškos algoritmų, kuriems reikalingi surikiuoti pradiniai duomenys.

Šioje užduotyje aprašytas vadinamasis nykštukų rikiavimo algoritmas (angl. *gnome sort*), taip pat dar vadinamas elementariuoju (angl. *stupid sort*). Jis yra idėjiškai gana paprastas, tačiau dažniausiai gana neefektyvus. Tiesa, jei pradiniai duomenys yra beveik išrikiuoti, šiam algoritmui užtenka n žingsnių turint n elementų. Nykštukų rikiavimo algoritmą pasiūlė Irano informatikos ir inžinerijos profesorius Hamidas Sarbazi-Azadas 2000 metais.

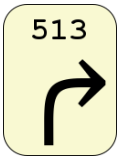
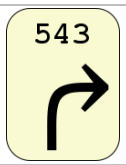
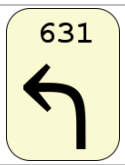
Kalbėdami apie algoritmus mes visuomet galvojame apie jų našumą, pavyzdžiui, kiek operacijų reikės blogiausiu atveju priklausomai nuo elementų kiekio. Nykštukų rikiavimo algoritme, jei turime n rąstų, blogiausiu atveju mums reikės apytiksliai n^2 operacijų. Tai vadinama kvadratiniu sąryšiu, nusakančiu šio algoritmo sudėtingumą. Yra keletas kitų rikiavimo algoritmų, kurių sudėtingumas yra gerokai mažesnis nei kvadratinio sudėtingumo algoritmo.

55. Bebrų ralis

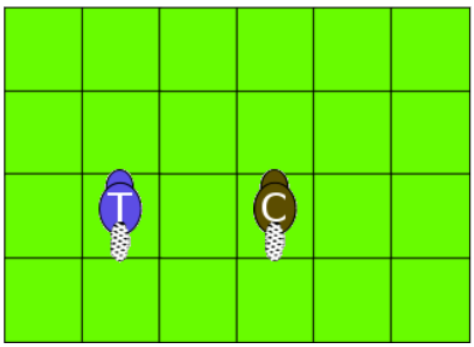
Cecilija ir Tadas žaidžia stalo žaidimą „Bebrų ralis“. Kiekvienas žaidėjas turi po bebrą, kurio judėjimas nusakomas žaidimo kortelėmis. Kortelėse nurodytas judesys: pasisukti į dešinę ar kairę (liekant tame pačiame langelyje) arba judėti vieną langelį pirmyn.



Žaidėjai pasirenka 4 korteles ir išdėsto jas taip, kaip nori, kad jų bebras judėtų. Kiekviename iš keturių žaidimo etapų abu žaidėjai vykdo po vieną kortelėje pavaizduotą judesį. Judesys kortelėje su didesniu pirmenybės skaičiumi vykdomas pirmas. Jei bebras juda į užimtą langelį, jis stumia kitą bebrą į šalia esantį langelį judėjimo kryptimi.

	1	2	3	4
 Cecilijos kortelės	 231	 321	 136	 513
 Tado kortelės	 123	 543	 564	 631

Nuvilkite bebrus į langelius, kuriuose jie atsidurs po keturių žaidimo etapų.



Paaiškinimas

1. Pirmas judesys gali būti vykdomas vienu metu. Bebrai juda vienu langeliu pirmyn.

2. Antras judesys taip pat gali būti vykdomas vienu metu. Bebrai atliks posūkius.

3. Trečiu žingsniu abu bebrai turėtų judėti į tą patį langelį. Tado bebras gali judėti pirmas, kadangi jo kortelė turi didesnį prioriteto skaičių (564). Tik tuomet Cecilijos bebras judės vieną žingsnį pirmyn ir pastums Tado bebrą.

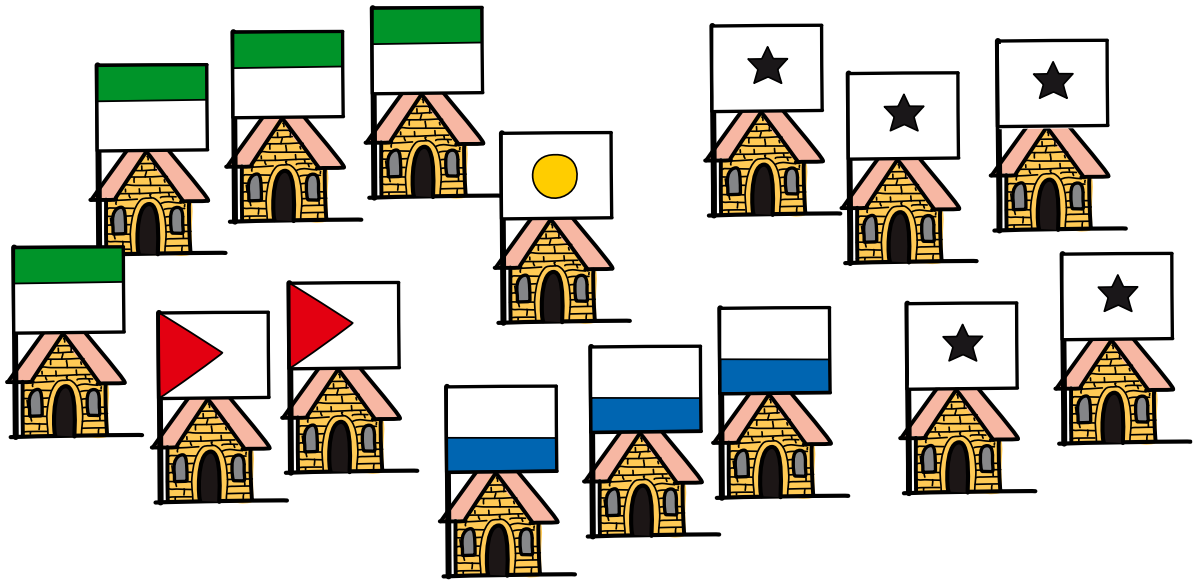
4. Paskutiniu žingsniu abu bebrai pasisuks vienu metu.

Tai informatika!

Uždaviniai dažnai skaidomi į mažesnes dalis. Tai leidžia keletui procesų vykti nepriklausomai ir vienu metu spręsti keletą mažesnių uždavinių. Tai vadinama lygiagrečiuoju procesų vykdymu. Vykdant procesus vienu metu gali atsitikti taip, kad vienas procesas turi palaukti, kol gaus prieigą prie bendrų išteklių. Tam svarbu apibrėžti procesų vykdymo koordinavimo taisykles. Šiame uždavinyje – tai pirmenybės skaičiai judesių kortelėse ir kito bebro postūmis į artimiausią langelį.

56. Bebrijos susivienijimas

Kadaise Bebrijos šalyje gyveno penkios karingos gentys. Kiekviena gentis turėjo po keletą namų, kaip parodyta paveiksle.



Gentys nusprendė susivienyti. Nutarta, jog vienu metu vienytis gali tik dvi gentys.

Dvi susivienijusios gentys kiekviename iš savo namų švęsdavo po savaitę, tad dviejų genčių vienijimasis trukdavo tiek savaičių, kiek besivienijančiose gentyse buvo namų. Tik po to buvusios dvi gentys tapdavo viena gentimi ir galėdavo tęsti vienijimąsi, kol iš viso liks tik viena bendra gentis.

Kiek mažiausiai savaičių reikėjo visoms gentims susivienyti?

A. 15

B. 33

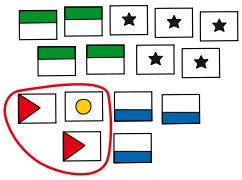
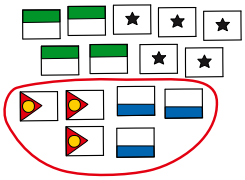
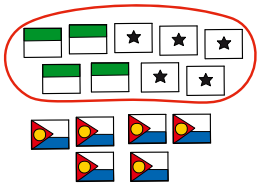
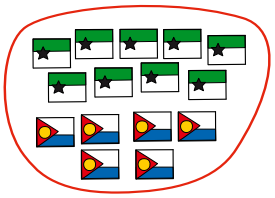
C. 50

D.120

Paaiškinimas

Teisingas atsakymas: B (reikėjo 33 savaičių).

Norint kiek įmanoma sumažinti visų genčių susivienijimui reikalingų savaičių skaičių reikia, kad kiekvienas namas kuo mažiau kartų būtų įtraukiamas į vienijimosi procesą. Iš čia aišku, kad didžiausios gentys turėtų vienytis paskutinės, nes tik taip didžiausias namų skaičius į vienijimosi procesą bus įtrauktas mažiausiai kartų. Taigi, kiekviename vienijimosi žingsnyje turėtų dalyvauti dvi mažiausiai namų turinčios gentys. Žingsnis po žingsnio šis procesas parodytas šioje lentelėje; aiškumo dėlei skliausteliuose nurodomas tik genčių dydis:

(1) ir (2) susijungia į gentį (3), tai trunka tris savaites.	(3) ir (3) per šešias savaites susijungia į gentį (6).	Dabar (4) ir (5) susijungia į gentį (9) per devynias savaites.	Galiausiai (6) ir (9) susijungia, tai užtrunka 15 savaičių.
			

Visas jungimosi procesas baigiamas po 15 + 9 + 6 + 3 = 33 savaičių, kaip ir yra atsakymo variante B.

Pateikta susivienijimų seka nėra vienintelė, leidžianti rasti optimalų sprendimą. Pradžioje taip pat būtų galima sujungti (4) ir (5) ir tik tada sujungti (1) ir (2), kurių rezultatas (3). Galiausiai bet kuriuo atveju sujunkite kylančias gentis (6) ir (9).

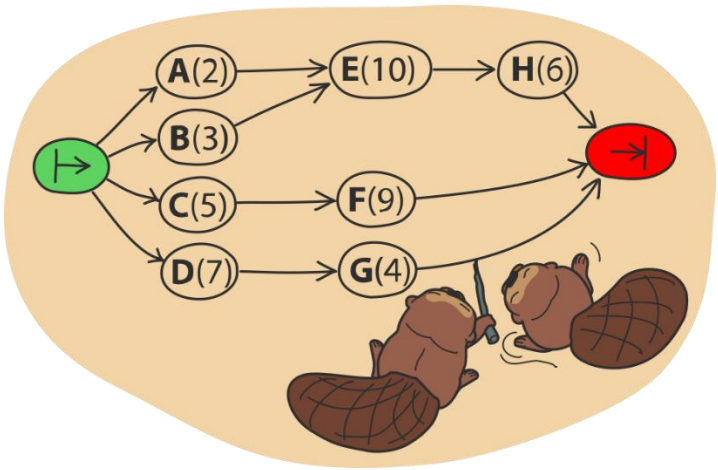
Tai informatika!

Tai optimizavimo uždavinio pavyzdys – tokių uždavinių tikslas yra kiek įmanoma sumažinti arba padidinti kokį nors dydį, atsižvelgiant į tam tikrus ribojimus. Optimizavimo uždaviniai dažnai pasitaiko ir kasdieniniame gyvenime: rasti trumpiausią maršrutą, sudaryti tvarkaraštį, kuriame veiklos kuo mažiau persidengtų ir pan. Yra keletas būdų optimizavimo uždaviniams spręsti, o vienas iš jų – godusis algoritmas.

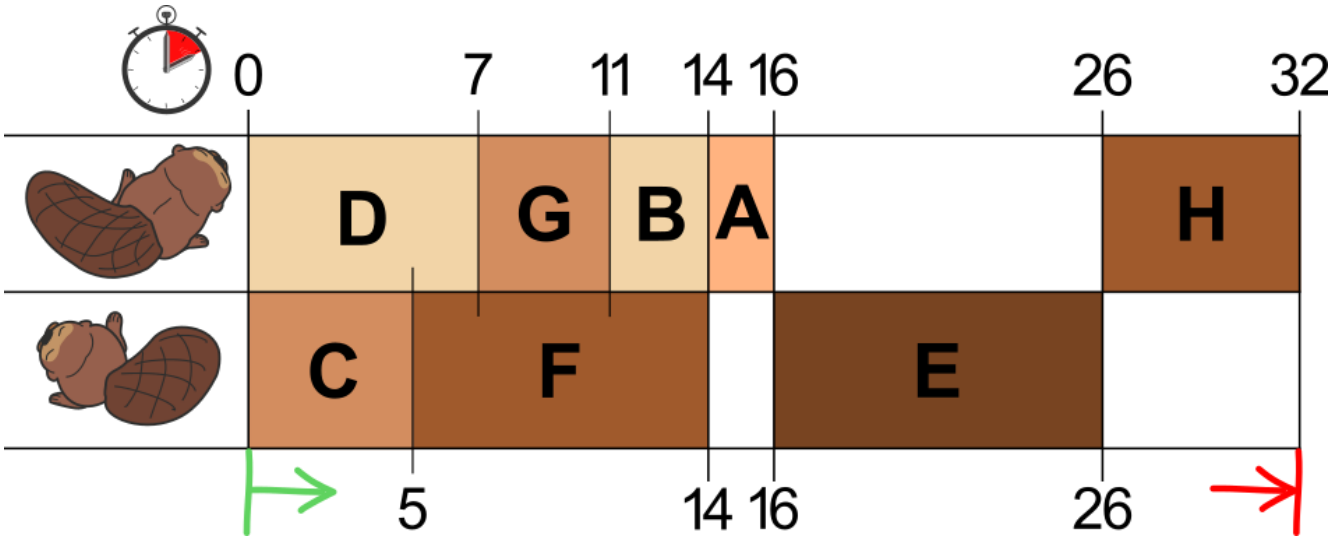
Godusis algoritmas remiasi prielaida, kad geriausias pasirinkimas kiekviename atskirame žingsnyje duos geriausią galutinį rezultatą. Tokia prielaida išlaikoma ir šiame uždavinyje: gentys turi kiek įmanoma sumažinti vienijimosi trukmę kiekviename žingsnyje, kad bendra vienijimosi trukmė būtų kuo trumpesnė. Vis dėlto, ne visus optimizavimo uždavinius galima spręsti taikant godųjį algoritmą, tačiau dažniausiai šiuo algoritmu pavyks rasti pakankamai gerą sprendimą per pakankamai trumpą laiką.

57. Du bebrai dirba drauge

Du bebrai stato užtvanką ir turi atlikti 8 užduotis (nukirsti medžius, pašalinti šakas, surinkti kamienus ir pan.): A(2), B(3), C(5), D(7), E(10), F(9), G(4), H(6). Skaičius skliaustuose nurodo, kiek valandų reikia užduočiai atlikti. Kai kurias užduotis galima atlikti tik užbaigus ankstesnes, tai paveiksle nurodoma rodyklėmis. Bebrai darbuojasi vienu metu, tačiau kiekvienas imasi atskiros užduoties.



Bebrai naudoja tokią strategiją: iš galimų užduočių pasirenka ilgiausiai trunkančią. Užduočių atlikimo planas atrodytų šitaip:



Bebrai pastatytų užtvanką per 32 valandas. Taikant kitokią strategiją užtvanką galima būtų pastatyti per trumpesnį laiką.

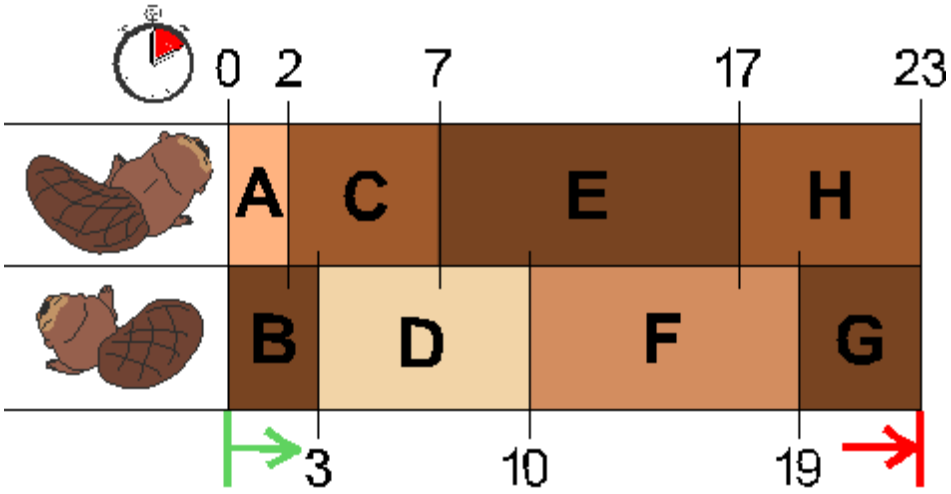
Kiek mažiausiai valandų reikia dviem bebrams užtvankai pastatyti?

Paaiškinimas

Teisingas atsakymas: 23.

Uždavinio lentelėje parodytas dviejų bebrų užduočių atlikimo planas. Matome, kad pirmasis bebras neturi darbo gana ilgą laiką, net 8 valandas, o antrasis – 6 valandas. Geriausia būtų, jei jie galėtų darbuotis visą laiką.

Taikysime kitą strategiją: planuosime, kad dvi ilgiausiai trunkančios užduotys E(10) ir F(9) neatitektų tam pačiam bebrui. Pateikiame vieną iš tokių užduočių paskirstymo planų.



Matome, kad užtvanką galima pastatyti net per 23 valandas!

Pastebėkime, kad ši strategija leido atlikti darbą per trumpiausią laiką todėl, kad abu bebrai galėjo darbuotis nedarydami pertraukų.

Tai informatika!

Kai kuriems uždaviniams bebrų strategija (imk ilgiausiai trunkančią užduotį iš likusių) gali pateikti optimalų sprendimą – užduotys gali būti atliktos per trumpiausią laiką. Kitiems uždaviniams (kaip čia pateiktas) geriau tinka ilgiausiai trunkančių užduočių paskirstymas skirtingiems atlikėjams. Deja, kiekvienai šiai strategijai galima rasti uždavinių pavyzdžių, kuriems ji netinka. Taip yra todėl, kad šiems uždaviniams nėra bendro sprendimo trumpiausiai laiko trukmei rasti – optimalų sprendimą galima rasti tik išbandant visus galimus užduočių paskirstymo variantus! Paprastai tai nėra realu ir net skaičiuojant kompiuteriu procesas gali trukti ilgiau nei vienas bebras pastatytų visą užtvanką!

Šio uždavinio pavyzdys buvo kruopščiai parinktas, kad bebrų strategija (imk ilgiausiai trunkančią užduotį iš likusių) netiktų. Tai nebuvo lengva. Daugumai uždavinių ši bebrų strategija (dažnai vadinama godžiąja) pakankamai gerai tinka, galima greitai paskirstyti užduotis ir bebrai gali imtis darbuotis nieko nelaukdami.

Sukurti pavyzdžių, netinkančių kuriai nors sprendimo strategijai (kaip buvo daroma rengiant šį uždavinį), reikalauja kūrybiškumo ir gilaus strategijos suvokimo. Tačiau šie įgūdžiai reikalingi informatikui norint, pavyzdžiui, nustatyti ilgiausią programos atlikimo kompiuteriu laiką, kas vartojama algoritmų analizėje (skaičiavimų sudėtingumo teorijos srityje).

https://en.wikipedia.org/wiki/Scheduling_(computing)

https://en.wikipedia.org/wiki/Topological_sorting

https://en.wikipedia.org/wiki/Greedy_algorithm

https://en.wikipedia.org/wiki/Computational_complexity_theory

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 uždavinių rinkinys tinka Mokyklos pedagogikos studijų programos moduliui „Informatikos didaktika“. Studentai, būsimi mokytojai, nagrinėdami „Bebro“ uždavinius susipažįsta su įvairiais informatikos konceptais, nagrinėja sudėtingesnes informatikos sąvokas, išmoksta jas paaiškinti .