ISSN 2783-7599

b·e·b·r·a·s

Informatikos ir informatinio mąstymo uždavinių rinkinys

Nr. 4

ŠVIETIMO, MOKSLO IR SPORTO MINISTERIJA

VILNIAUS UNIVERSITETAS · 1579 · UNIVERSITAS VILNENSIS

Bebras
International Challenge on Informatics and Computational Thinking

Šiame rinkinyje pateikiami XV informatikos ir informatinio mąstymo konkurso „Bebras“ II etapo uždaviniai, jų atsakymai ir paaiškinimai, koks informatikos turinys ir konceptai atskleidžiami, kaip ir kuo uždavinys ypatingas ar įdomus informatikai. Visi uždaviniai (įskaitant grafiką ir kitą medžiagą) licencijuojami pagal Kūrybinių bendrijų licenciją – „Creative Commons Attribution-ShareAlike 4.0 International License“. Šis uždavinių rinkinys skirtas ugdyti 9–12 klasių mokinių informatinio mąstymo gebėjimus.

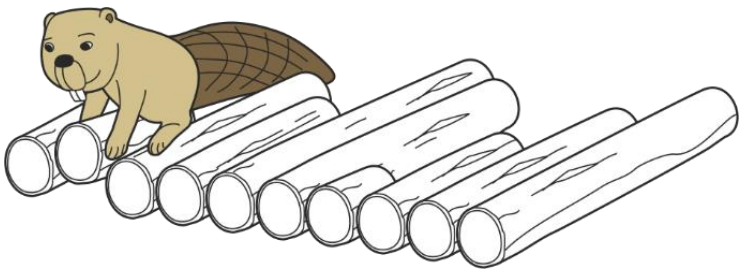
Dėkojame Daumilui Ardickui, dr. Eglei Jasutei, dr. Tatjanai Jevsikovai, dr. Eimučiui Karčiauskui, Audronei Klupšaitei, Alvidai Lozdienei, Modestui Rimkui, dr. Broniui Skūpui, dr. Gabrielei Stupurienei, Tomui Šiauliui, talkinusiems verčiant ir adaptuojant uždavinius. Taip pat dėkojame tarptautinei „Bebro“ konkurso bendruomenei ir uždavinių autoriams.

Parengė Lina Vinikienė
Konsultavo Valentina Dagienė
Redagavo Viktoras Dagys
Viršelį kūrė Vaidotas Kinčius

Įvadas

„Bebras“ – tai ne tik konkursas (anglų k. jis vadinamas „challenge“ – iššūkiu). Tai daugybė įvairių veiklų, kurios vyksta ištisus metus.

„Bebro“ konkurso tikslas – sudominti bendrojo ugdymo mokyklų mokinius informatikos mokslo fundamentinėmis idėjomis, skatinti sumaniau naudotis informacinėmis technologijomis, ugdyti mokinių kūrybiškumą, informacinę kultūrą, algoritminį, loginį, kritinį ir informatinį mąstymą.



2018 metų „Bebro“ konkurso uždavinius pasaulyje sprendė per 2 780 000 mokinių iš 57 valstybių. Lietuvoje tais metais dalyvavo 44 247 mokiniai.

„Penkiolika metų lyg ir trumpas laikotarpis, tačiau mums džiugu, kad jau tiek metų esame drauge, kad mokiniai sprendžia uždavinius mokykloje, dalinasi idėjomis su draugais, mokytojais. Džiugu, kad turime „Bebro“ konkurso idėją, kuri mus vienija. Pirmasis konkursas prieš 15 metų įvyko tik Lietuvoje, o dabar jis vyksta daugiau nei 60 šalių. Informatinis mąstymas – tai nėra tik informatika, tai ir mokėjimas analizuoti, spręsti uždavinį, abstrahuoti. „Bebro“ uždaviniai trumpi, tačiau juose slypi galingos idėjos“ – sako konkurso sumanytoja Vilniaus universiteto profesorė Valentina Dagienė.

Pasak Vilniaus universiteto studijų prorektoriaus dr. Valdo Jaskūno, „Bebras“ įkūnija tai, ko siekia Vilniaus universitetas perimdamas mokytojų ugdymą – novatorišką, patrauklų mokymą ir mokymąsi. „Bebras“ kaip konkursas, kaip judėjimas, kaip iniciatyva yra turbūt vienas iš labiausiai matomų Vilniaus universiteto ambasadorių pasaulyje.[1]

XV informatikos ir informatinio mąstymo konkurso „Bebras“ II etapas vyko 2019 m. vasario 2 d. Jame dalyvavo 259 jauniai (9–10 klasių mokiniai) ir 247 kolegos (11–12 klasių mokiniai).

[1] Vilniaus universitetas: https://naujienos.vu.lt/15-asis-konkurso-bebras-gimtadienis/

2018 m. „Bebro“ konkurso dalyvių skaičius pagal šalis (57 šalių duomenys)

Lentelėje pateikiamas XV konkurso II etapo uždavinių skirstymas pagal amžiaus grupes.

Nr.	Pavadinimas	Uždavinio identifikatorius	Jauniai (9-10 kl.)	Kolegos (11-12 kl.)
1	Bebrų tunelis	2016-CH-04a	6	
2	Raudoni ir mėlyni rutuliukai	2016-IT-02b	6	6
3	Slaptažodis	2017-PT-02b	6	
4	Įvairūs pomėgiai	2015-AT-01	6	
5	Pranešimai vėliavėlėmis	2018-TW-06	6	
6	Knygų rinkinio rikiavimas	2018-CN-02	9	6
7	Pramiegojimo kaina	2018-IR-03	9	6
8	Ilgiausia žodžių grandinė	2018-CZ-08c	9	9
9	Mokami keliai	2018-IR-06	9	9
10	Šiurpnakčio pasivaikščiojimas	2018-SK-04	9	
11	Robotas-sodininkas	2018-CY-05	12	9
12	Savanaudės voverės	2016-RU-08	12	12
13	Lobio žemėlapiai	2018-TW-04	12	12
14	Įjunkite šviesas!	2018-DE-06	12	
15	Ryšių tinklai	2018-RO-07	12	
16	Masonų šriftas	2018-LT-10		6
17	Dykuma	2018-HR-08		6
18	Paskyros kūrimas	2018-IR-01		9
19	Žemėlapio žaidimas	2018-TW-07		9
20	„Soundex“ algoritmas	2018-PK-06		12
21	Sesės ir brolio paslaptys	2018-UK-04		12
22	Šviesolaidžių tinklas	2018-SP-02		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. Bebrų tunelis

Bebrų šeima pasiekė siaurą, tamsų tunelį. Vienas ar daugiausia du bebrai gali eiti tuneliu vienu metu ir tik pasišviesdami žibintuvėliu. Šeimos nariai tunelį įveikia skirtingai: sūnus Benas užtrunka 5 minutes, sesuo Ona sugaišta trigubai ilgiau, mamai prireikia 30 minučių, o tėčiui – 35 minučių. Bebrų šeima turi tik vieną žibintuvėlį.

Kiek trumpiausiai laiko truktų kelionė, kol visi keturi bebrai atsidurtų kitoje tunelio pusėje?

Paaiškinimas

Svarbu pastebėti: skaičiai parinkti taip, kad du lėčiausi bebrai – mama ir tėtis – turi eiti tuneliu kartu ir bent vieno iš jų grįžimas užimtų per daug laiko (tai truktų 90 ar 100 minučių). Sprendimas būtų toks: du greičiausi bebrai – Ona ir Benas – eina pirmi (15 minučių), vienas iš jų grįžta atgal (5 ar 15 minučių), perduoda žibintuvėlį tėčiui ir mamai, kurie pereina tunelį kartu (35 minutės), paskui kitas jaunasis bebras grįžta (15 ar 5 minutės) pas likusį bebrą ir abu kartu vėl pralenda pro tunelį (15 minučių). Visa tai užima 85 minutes.

Yra du sprendiniai:

Kelias	Pirmas	Antras	Minutės
1: Eina tuneliu	Ona	Benas	10
2: Grįžta atgal	Benas		15
3: Eina tuneliu	Mama	Tėtis	40
4: Grįžta atgal	Ona		50
5: Eina tuneliu	Ona	Benas	60

Kelias	Pirmas	Antras	Minutės
1: Eina tuneliu	Ona	Benas	10
2: Grįžta atgal	Ona		20
3: Eina tuneliu	Mama	Tėtis	45
4: Grįžta atgal	Benas		50
5: Eina tuneliu	Ona	Benas	60

Tai informatika!

Informatikai dažnai susiduria su įvairiais ribojimais. Dažnai tai būna susiję su laiko optimizavimu ar su optimalaus sprendimo paieška.

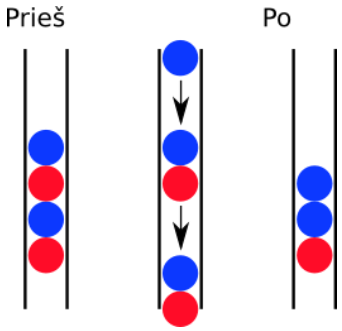
Šiame uždavinyje bebrų šeimai reikia kaip įmanoma greičiau (optimizavimas) pereiti tunelį, tačiau vienu metu tunelyje daugiausia gali būti du bebrai (ribojimas).

Besimokydami pagrindų, informatikai įgyja patirties, kaip spręsti tokias problemas ir rasti optimalų sprendimą.

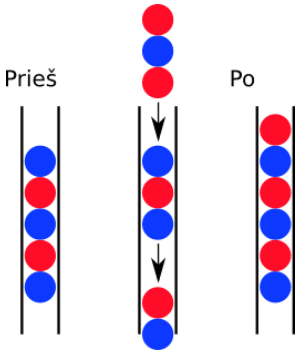
2. Raudoni ir mėlyni rutuliukai

Emilis sprendžia galvosūkį kompiuteriu. Jis sudeda į cilindrą ne mažiau kaip tris rutuliukus. Rutuliukai yra vienspalviai: raudoni ir mėlyni. Paspaudus mygtuką PIRMYN du apatiniai rutuliukai iškrenta žemyn. Cilindras pasipildo naujais rutuliukais priklausomai nuo pirmojo iškritusio rutuliuko spalvos.

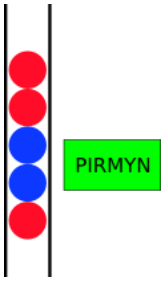
Jei pirmasis rutuliukas raudonas, tai iš viršaus į cilindrą įrieda naujas mėlynas rutuliukas.



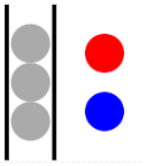
Jei pirmas rutuliukas mėlynas, tai iš viršaus į cilindrą įrieda trys nauji rutuliukai: raudonas, mėlynas, raudonas.



Jei cilindre yra bent trys rutuliukai, Emilis spaudžia mygtuką PIRMYN. Tai kartoja tol, kol cilindre lieka du ar mažiau rutuliukų – tada žaidimas baigiamas. Pavyzdžiui, pateiktoje situacijoje paspaudus mygtuką penkis kartus cilindre liks tik du mėlyni rutuliukai ir žaidimas bus baigtas.



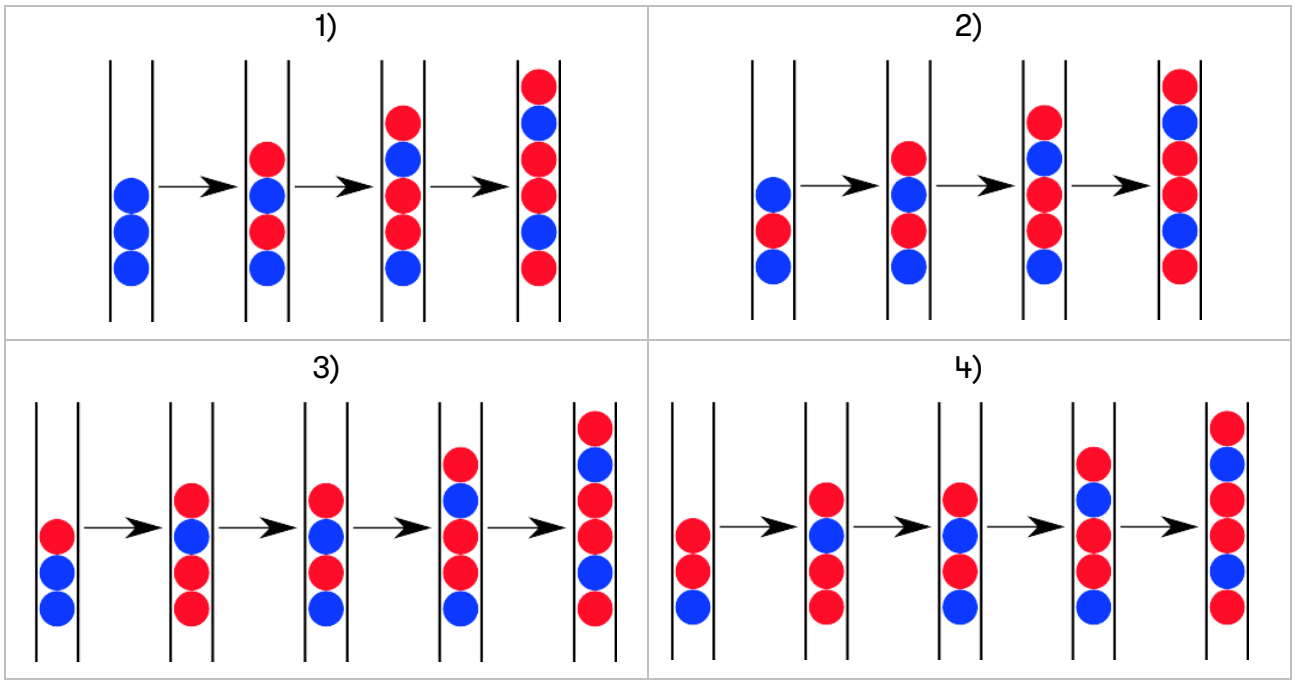
Tempdami dešinėje esančius rutuliukus į cilindrą sudarykite trijų rutuliukų seką taip, kad žaidimas niekada nesibaigtų.



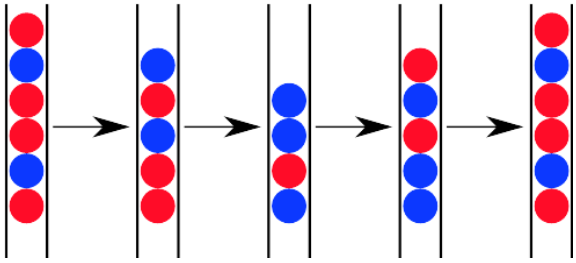
Paaiškinimas

Žaidimas niekada nesibaigs, jei Emilis pradės kurti dėklą mėlynu rutuliuku, kuris atsidurs apačioje. Jei pradinio dėklo iš trijų rutuliukų apačioje bus raudonas, tai paspaudus mygtuką vieną kartą žaidimas bus baigtas. Tačiau jei apatinis rutuliukas bus mėlynas, tai daugiausiai po penkių paspaudimų, nepriklausomai nuo likusių, bus suformuotas toks dėklas: RMRRMR.

Jei sekoje pirmas rutuliukas vaizduoja viršutinį, tai gaunami tokie rezultatai:



Tada žaidimas cikliškai kartojasi kas keturis mygtuko PIRMYN paspaudimus:



Tai informatika!

Šis galvosūkis vaizduoja *Post production* sistemą, skaičiavimų modeliu naudojamas eilučių perrašymas, kurį 1920 m. sukūrė Emilis Leonas Postas (pirmą kartą tai buvo paskelbta 1943 m.). E. L. Postas (1897–1954) buvo lenkų kilmės JAV gimęs matematikas ir logikas.

Perrašinėjimo modeliai apima įvairias formulių sistemas, kurios leidžia pakeisti vieną eilutės dalį kita. Perrašinėjimo modelis gali būti ir objektų sistema kartu su seka ryšių, lemiančių galimus manipuliavimus su šiais objektais ir jų transformacijas.

Teorinis kompiuterių mokslas šias sistemas laiko kalbomis be konteksto. Pavyzdžiui, daugybos ir sudėties sistemos gali būti apibrėžiamos vos keliomis taisyklėmis, jei naudojama kalba be konteksto. Kitas žinomas kalbos be konteksto pavyzdys yra tikslus ir išsamus programavimo kalbos apibrėžimas, kuris naudotinas ir edukaciniams tikslams, ir skaičiavimų įrenginiuose.

3. Slaptažodis

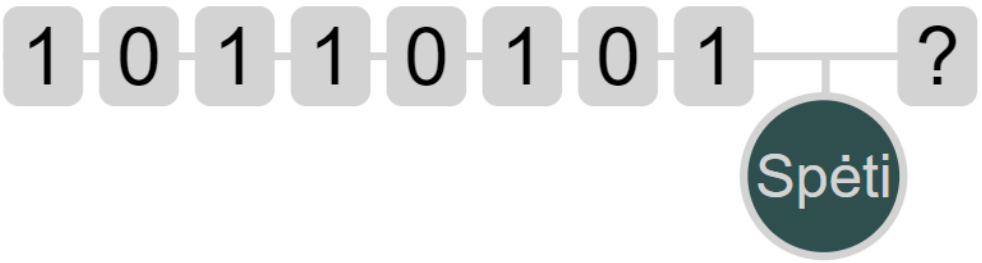
Bebras kaupia tvirtus rąstus vėlesniam naudojimui ir juos rakina buveinėje. Spyna užrakinama 8 dvejetainių (tik 0 ir 1) skaitmenų ilgio slaptažodžiu. Kartą bebras atvilko puikių klevo ir gluosnio rąstų, užrakino ir pamiršo slaptažodį.

Bebro spyna išmanioji: yra ekranas įvedamiems 8 slaptažodžio skaitmenims rodyti, spėjimo mygtukas ir atsako langelis. Įvedus bet kokį slaptažodį ir paspaudus spėjimo mygtuką, atsako langelis rodys štai ką:

- Jei teisingai atspėti lygiai 4 skaitmenys, bus matomas skaičius 4;
- Jei teisingai atspėti visi 8 skaitmenys, bus matomas skaičius 8;
- Bet kuriuo kitu atveju bus matomas brūkšnelis „-“.

Pavyzdžiui, jei bebras spėjo 10110111, o teisingas slaptažodis yra 11010101, tai atsako langelyje matysime „-“. Tačiau jei bebras įvedė 10110110, tai atsako langelyje matysime skaičių 4, o jei įvedė 11010101, tai matysime skaičių 8.

Rodomas toks bebro spynos ekrano vaizdas:



Padėk bebrui atspėti slaptažodį.

Paaiškinimas

Pateikę bet kokį slaptažodį ir neatspėję, toliau bandome jį šiek tiek modifikuoti ir stebime rezultatą. Tarkim, pirmąkart pateikę atspėjome 4 teisingus skaitmenis. Tada invertuojame pirmą skaitmenį (0 keičiame 1 arba atvirkščiai – 1 keičiame 0) ir vėl spėjame. Jei naujo spėjimo rezultatas yra 5 teisingi skaitmenys, vadinasi, spėjimą pagerinome ir jau atspėjome pirmąjį skaitmenį. Tačiau jei naujas spėjimas pateikia 3 teisingus skaitmenis, vadinasi, pirmasis skaitmuo buvo geras ir nereikėjo jo keisti. Jokio kito rezultato negausime, nes vieno skaitmens pakeitimas teisingų skaitmenų skaičių padidina arba sumažina vienetu.

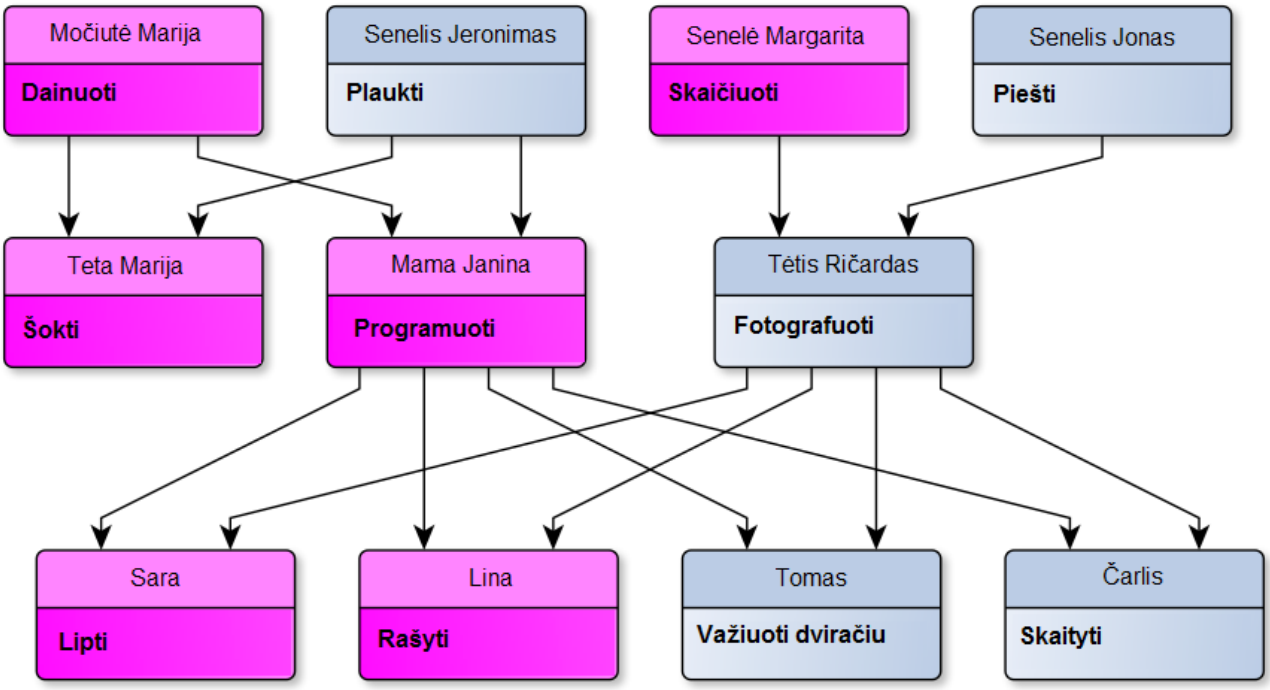
Šią procedūrą kartojame, norėdami išsiaiškinti likusius 7 skaitmenims, ir taip sudarome teisingą slaptažodį.

Tai informatika!

Dvejetainė skaičiavimo sistema yra kompiuterio veikimo ir skaičiavimų pagrindas. Kompiuteris bet kokią informaciją išreiškia dviem būsenomis. Šiame uždavinyje taip pat svarbu pastebėti, kaip pateikti grįžtamąjį ryšį (rezultatą). Pateikę pradinį duomenį ir gavę rezultatą, galime geriau pasirinkti tolesnius spėjimus.

4. Įvairūs pomėgiai

Kiekvienas Linos šeimos narys turi savo pomėgį. Dukra paveldi pomėgius iš mamos, o sūnus – iš tėvo. Be paveldėtų pomėgių, kiekvienas šeimos narys turi dar po vieną savo pomėgį. Paveikslėlyje vaizduojami Saros, Linos, Tomo ir Čarlio pomėgiai, taip pat jų protėvių pomėgiai.



Iš paveikslėlio matyti, kad mama Janina pomėgį dainuoti paveldėjo iš močiutės Marijos. Be šio pomėgio, ji dar mėgsta programuoti. Lina iš mamos paveldėjo abu šiuos pomėgius, be to, ji dar mėgsta rašyti. Taigi Lina geba dainuoti, programuoti ir rašyti.

Kuris iš pateiktų teiginių yra teisingas?

1. Sara mėgsta rašyti, programuoti ir dainuoti.
2. Tomas paveldėjo pomėgį plaukti iš savo senelio Jeronimo.
3. Teta Marija mėgsta šokti ir plaukti.
4. Tomas mėgsta važiuoti dviračiu, piešti ir fotografuoti.

Paaiškinimas

4-as teiginys teisingas, nes Tomas paveldi gebėjimą piešti iš senelio ir gebėjimą fotografuoti iš tėčio.

1-as teiginys neteisingas, nes Sara nepaveldi gebėjimo rašyti iš sesers Linos.

2-as teiginys neteisingas, nes Tomas nepaveldi gebėjimų iš močiutės.

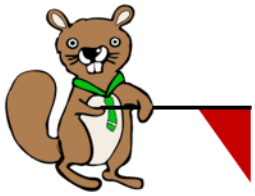
3-as teiginys neteisingas, nes teta Marija nepaveldi gebėjimo plaukti iš jos tėčio Jeronimo.

Tai informatika!

Paveldimumas yra svarbi objektinio programavimo koncepcija. Programinės sistemos dalys gali būti pakartotinai panaudojamos bei išplečiamos. Šiuo atveju sugebėjimai yra programinės sistemos dalys, kurios yra paveldimos ir papildomos kitais sugebėjimais. Šioje užduotyje paveldimumas skiriasi nuo programavimo, nes ne viskas yra paveldima.

5. Pranešimai vėliavėlėmis

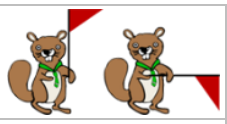
Vėliavų mieste bebrai bendrauja vėliavėlėmis. Sutarta, kad vėliavėlės turi dvi būsenas: horizontali ir vertikali.



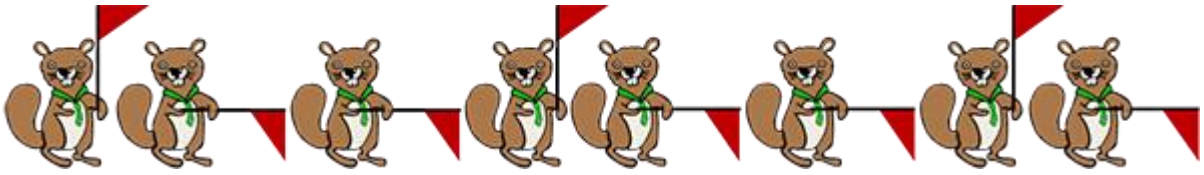
1 būsena: horizontali

2 būsena: vertikali

Matome, kaip nuosekliai keičiant vėliavėlės būsenas gaunamos penkios skirtingos raidės: P, Q, R, S, T.

				
P	Q	R	S	T

Bebrė Agota siunčia pranešimą:



Kurią raidžių seką išsiuntė Agota?

A. TSQ

B. RPQSR

C. RPSP

D. QPPTP

Paaiškinimas

Teisingas atsakymas: (D) QPPTP.

Horizontali būsena gali būti koduojama 0, o vertikali – 1. Tokiu atveju, Agotos pranešimo būsenų seka – vertikali, horizontali, horizontali, vertikali, horizontali, horizontali, vertikali, horizontali – koduojama taip: 10010010.

Raidės P kodas 0, raidės Q – 1, raidės R – 10, raidės S – 001, raidės T – 1001. Todėl D atsakymas yra teisingas: 1(Q)0(P)0(P)1001(T)0(P).

(A) TSQ --> 10010011 (klaida)

(B) RPQR --> 100100110 (klaida)

(C) RPSP --> 1000010 (klaida)

Tai informatika!

Seniau vėliavėlės buvo naudojamos jūreivystėje pranešimams perduoti (vadinamoji semaforo komunikavimo sistema). Siunčiant tokius pranešimus naudojamas kodas turi būti vienareikšmis. Šiuo atveju kodas – tai sistema, kai vėliavėlių būsenų vienareikšmiais rinkiniais koduojama kiekviena raidė. Vienas vienareikšmio kodo kūrimo būdų – priešdėlių naudojimas. Tai reiškia, kad nei vienos raidės kodas negali būti bet kurios kitos raidės prefiksu (priešdėliu).

Šioje užduotyje raidžių kodai turi prefiksus, nes P raidės kodas yra S raidės kodo prefiksas, Q raidės kodas – raidžių R ir T prefiksas, R raidės kodas – T prefiksas.

Šioje užduotyje kodas yra daugiaprasmis ir pranešimas gali būti koduojamas įvairiai: QPPQPPQP, QSSP, QPPTP, TSP, RPQSP, RPTP ir t. t. Tokiais atvejais pranešimai gali būti klaidingai suprantami.

Jei visų koduojamų objektų kodai vienodo ilgio, nelieka priešdėlių problemos.

Kodavimą ir dekodavimą nagrinėja informacijos teorija – viena iš informatikos sričių.

6. Knygų rinkinio rikiavimas

Trys bebrai turi po savo stalą. Ant kiekvieno stalo padėta po dvi knygas. Knygų tvarka sumaišyta, ir bebrai nori jas surikiuoti atlikdami apkeitimo žingsnius.

Yra du žingsnių tipai.

A tipo žingsnis: kiekvienas bebras gali sukeisti dvi ant stalo gulinčias knygas.

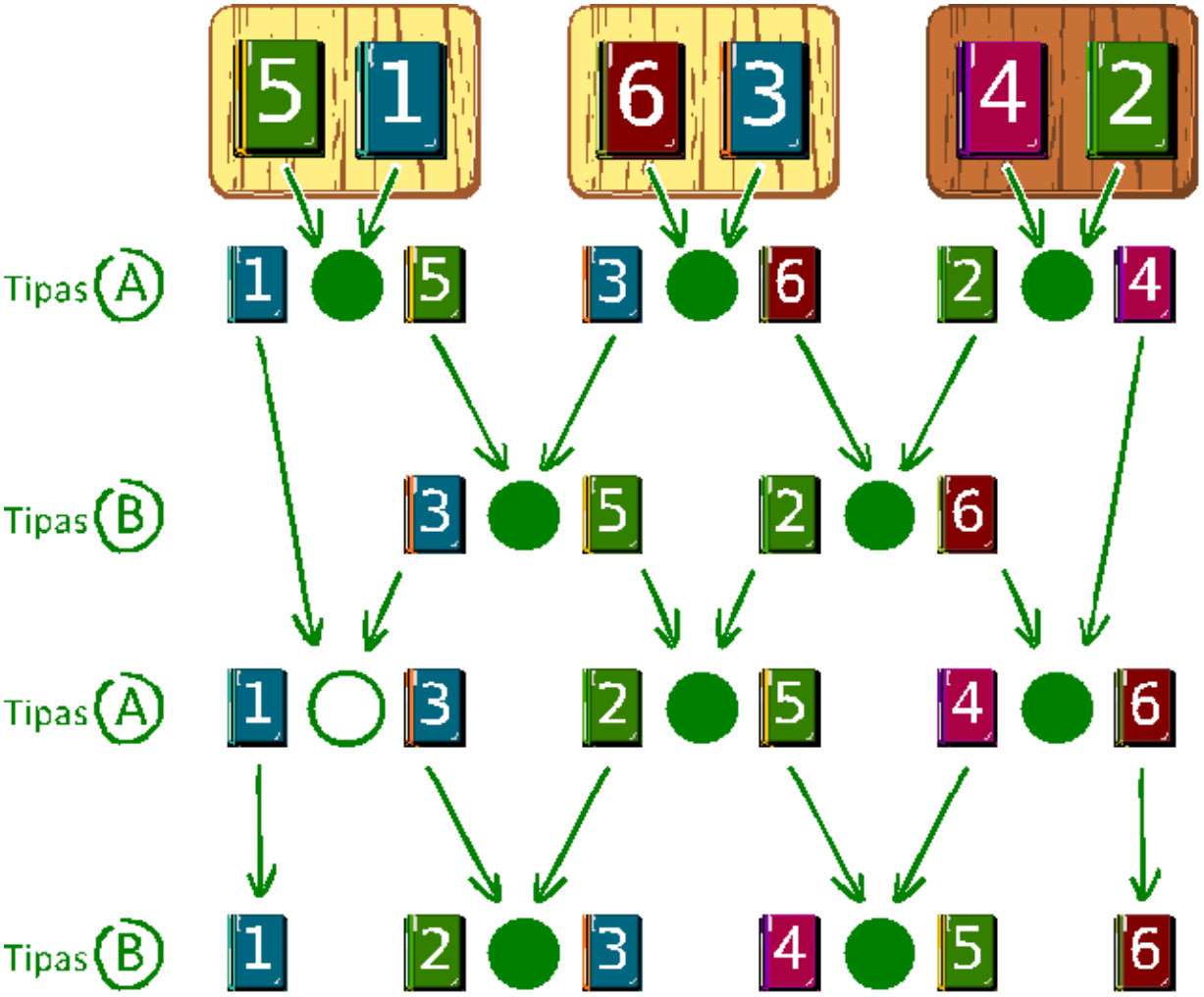
B tipo žingsnis: bebrai gali susikeisti kaimyninėmis knygomis su šalia esančiu stalu.

Kiek mažiausiai žingsnių reikia, kad knygos būtų išrikiuotos 1, 2, 3, 4, 5, 6?

A trijų žingsnių
B keturių žingsnių
C penkių žingsnių
D šešių žingsnių

Paaiškinimas

Teisingas atsakymas: B.



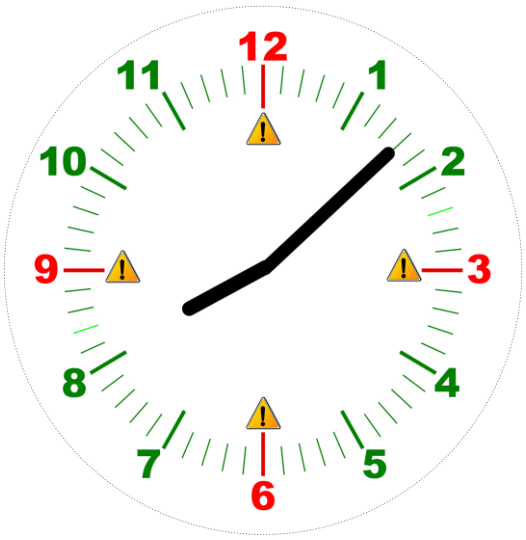
Šis uždavinys yra lygiagretaus tinklo rikiavimo algoritmo versija. Bebrai pradeda sukeisdami knygas ant kiekvieno stalo, kaip nurodyta uždavinyje. Remiantis godžiuoju algoritmu nusprendžiama, ar atlikti apkeitimą, ar ne. Antras žingsnis: vienintelis galimas veiksmas – visi bebrai stengiasi sukeisti knygas tarp stalų. Tuomet vienintelis kito žingsnio galimas veiksmas – sukeisti knygas ant kiekvieno stalo ir t. t. Godusis algoritmas yra intuityvus ir optimalus. Vykdant algoritmą, galima lengvai rasti atsakymą. Pateiktame paveikslėlyje parodyti minėtos strategijos veiksmai.

Tai informatika!

Rikiavimas – objektų išdėstymas eile pagal kurį nors jų parametrą: skaičių – pagal jų dydį, žodžių – pagal raidžių rikiavimo eilę abėcėlėje ir pan. Rikiuoti galima dvejopai: didėjančiai arba mažėjančiai. Jeigu objektai skaidomi į kelias grupes, sakoma, kad jie rūšiuojami. Rūšiavimas gali būti panaudotas kaip pagalbinis rikiavimo veiksmas. Objektai surūšiuojami į grupes (rūšis), po to grupės sujungiamos, ir gaunama surikiuotų objektų eilė. Atkreipiame dėmesį, kad anglų kalboje neskiriamos rikiavimo ir rūšiavimo sąvokos. Į tai reikia atsižvelgti, verčiant programas.

7. Pramiegojimo kaina

Benas dirba miesto Centrinėje traukinių stotyje. Darbą pradeda 8.00 val. Už kiekvieną 15 minučių vėlavimą nustatyta 10 € bauda. Pavyzdžiui, jei Benas į darbą atvyks iki 8.15 val. (pvz., 8.11), tai baudos mokėti nereikia, bet jei atvyks 8.20, tai turės mokėti 10 €.



Šį rytą Benas pramigo ir į savo gyvenamosios vietos traukinių stotį atvyko 08.08 val. Traukinių tvarkaraštis štai toks:

Traukinys	*Grafikas*	*Kelionės iki Centrinės traukinių stoties trukmė*	*Bilieto kaina*
Vietinis	Nuo 6.00 Išvyksta kas 5 min.	**40 min.**	5 €
Priemiestinis	Nuo 6.00 Išvyksta kas 10 min.	**30 min.**	10 €
Greitasis	Nuo 7.00 Išvyksta kas 15 min.	**20 min.**	15 €
Ekspresas	Nuo 7.00 Išvyksta kas 20 min.	**12 min.**	20 €

Kuriuo traukiniu vykti Benui yra mažiausiai nuostolinga?

A) Vietiniu
B) Priemiestiniu
C) Greituoju
D) Ekspresu

Paaiškinimas

Teisingas atsakymas: B (priemiestiniu).

- Jei Benas važiuotų priemiestiniu traukinių, tai mokėtų mažiausiai, nes:
- važiuodamas vietiniu traukiniu į Centrinę stotį atvyktų 08.50 val., sumokėtų 5 € už bilietą ir 30 € (3x10 €) baudą už vėlavimą į darbą, iš viso 35 €;
- važiuodamas priemiestiniu traukiniu į Centrinę stotį atvyktų 08.40 val., sumokėtų 10 € už bilietą ir 20 € (2x10 €) baudą už vėlavimą į darbą, iš viso 30 €;
- važiuodamas greituoju traukiniu į Centrinę stotį atvyktų 08.35 val., sumokėtų 15 € už bilietą ir 20 € (2x10 €) baudą už vėlavimą į darbą, iš viso 35 €;
- važiuodamas ekspreso traukiniu į Centrinę stotį atvyktų 08.32 val., sumokėtų 20 € už bilietą ir 20 € (2x10 €) baudą už vėlavimą į darbą, iš viso 40 €.

Tai informatika!

Užduotyje pateikiama optimizavimo problema. Optimizavimo metodai ir algoritmai yra viena iš svarbių informatikos mokslo sričių.

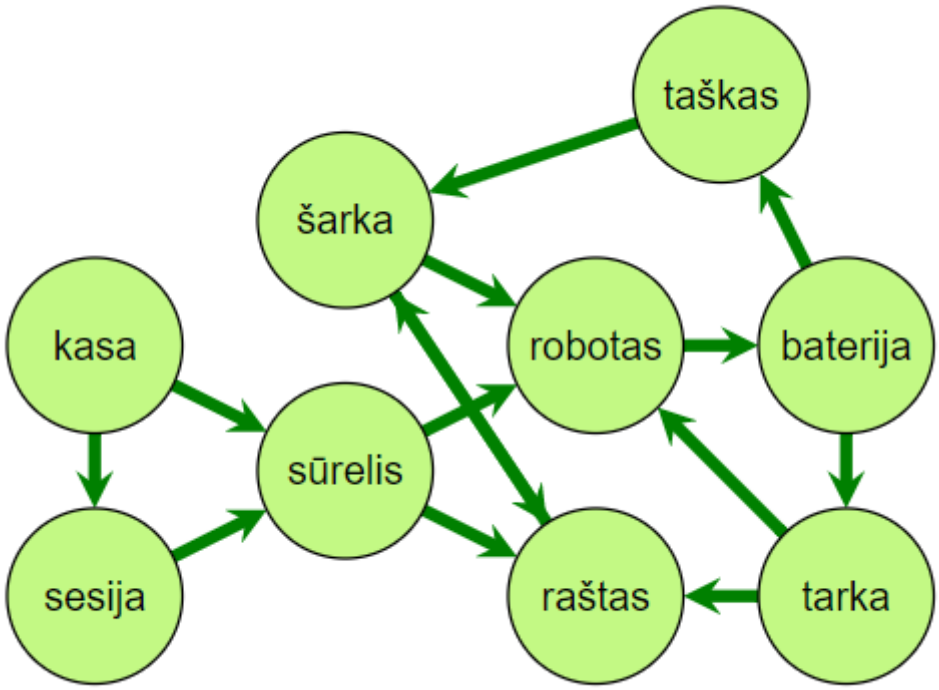
Optimizavimas informatikoje – tai programuotojo ar kompiliatoriaus atliekamas programinio kodo keitimas, siekiant pagreitinti programos veikimą, sumažinti naudojamą atmintį, sutrumpinti atlikimo laiką ar kitaip pagerinti, nekeičiant funkcionalumo.

Šaltinis: https://lt.wiktionary.org/wiki/optimizavimas

8. Ilgiausia žodžių grandinė

Du bebrai žaidžia žodžių grandinės žaidimą. Vienas bebras pradeda pasakydamas žodį. Tada kiekvienas paeiliui sako po dar nepanaudotą žodį, prasidedantį trečia ankstesnio žodžio raide.

Deja, bebrai žino labai nedaug žodžių – visą jų žodyną galima pavaizduoti taip:



Kiek žodžių sudaro ilgiausia žodžių seka, pasakyta per vieną žaidimą?

Paaiškinimas

Bebrai gali panaudoti daugiausiai 8 žodžius. Galima žodžių seka:

Kasa → sesija → sūrelis → robotas → baterija → tarka → raštas → šarka

Yra ir kitų tokio pat ilgio sekų. Turime pavyzdį iš 8 žodžių, bet dar nežinome, ar įmanoma sudaryti 9 žodžių seką. Tokiu atveju mums reikėtų panaudoti visus žodyno žodžius.

Kadangi žodyne nėra žodžio, kurio trečia raidė *k*, turėtume pradėti nuo žodžio „kasa“. Pabandę įvairius variantus pamatome, kad visais atvejais seka pasibaigia 8-uoju žodžiu. Tad neįmanoma viename žaidime panaudoti visų 9 žodžių.

Tai informatika!

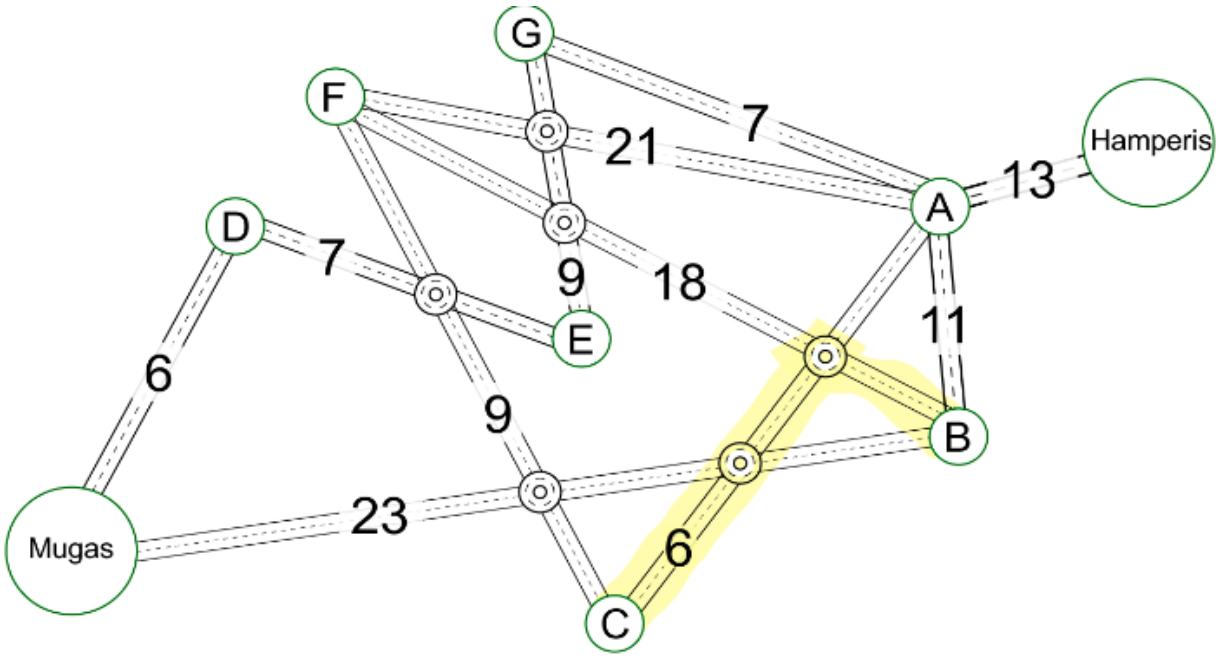
Spręsdami šį uždavinį turime suprasti taisyklių rinkinį, pasirinkti aiškų ir patogų duomenų vaizdavimą (grafą) ir tada surasti optimalų sprendimą duotoje sistemoje (žodyne). Grafai dažnai naudojami informatikoje. Jais modeliuojamos įvairiausios struktūros iš objektų ir jų ryšių, pavyzdžiui, traukinių maršrutų arba vandentiekio išdėstymo schemos.

Turėdami daugiau informatikos žinių, galėtume pakeisti uždavinį, pavyzdžiui, ilgiausio maršruto paieška grafe. Tai glaudžiai susiję su žinomu Keliaujančio pirklio uždaviniu (https://lt.wikipedia.org/wiki/Keliaujan%C4%8Dio_pirklio_u%C5%BEdavinys)

Informatikos mokslininkai teigia, kad surasti ilgiausią žaidimą didžiausiame žodyne (turint galvoje, kad žinome tūkstančius žodžių!) nėra įmanoma. Tai užimtų per daug laiko net su geriausiu algoritmu.

9. Mokami keliai

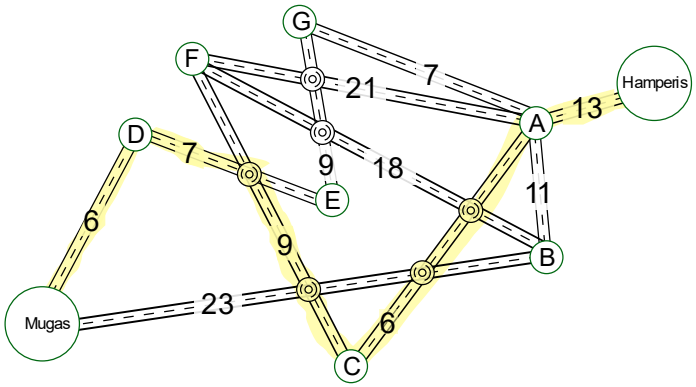
Benas nusprendė važiuoti iš Hamperio į Mugą. Žemėlapyje miestai žymimi apskritimais su raidėmis, linijos rodo dvipusio eismo kelius. Keliai gali kirstis, jų sankirtos pažymėtos skrituliais. Skaičius prie kelio rodo mokesčio sumą, kurią reikia sumokėti automobiliui įvažiuojant į šį kelią. Automobiliai gali keisti kryptį sankirtose, bet privalo sumokėti mokestį už kelią, į kurį įvažiuoja. Pavyzdžiui, iš B į C galima nuvykti važiuojant 18 ir 6 keliu ir tai kainuotų 24 = 18 + 6.



Kokį mažiausią kelių mokestį turi susimokėti Benas, kad nuvyktų iš Hamperio į Mugą?

Paaiškinimas

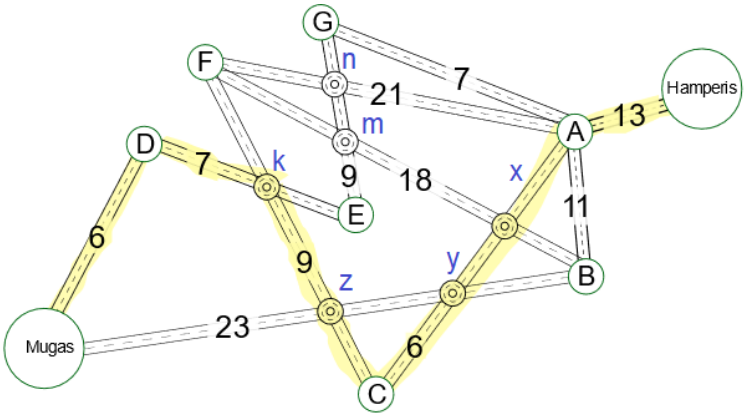
Paveiksle pažymėtas pigiausias maršrutas:



Norint įrodyti, kad šis maršrutas yra optimaliausias, būtina palyginti visus galimus variantus.

Tačiau galima peržvelgti kelius ir įsitikinti, kad reikia vengti brangių kelių: kelio iš B į Mugą, taip pat kelio iš A į F ir iš B į F.

Pažymėkime sankirtas.



Peržvelgiame maršrutus ir telieka patikrinti du maršrutus.

Galimi maršrutai	*Kaina*
Hamperis – A – C – k – D – Mugas	13 + 6 + 9 + 7 + 6 = 41
Hamperis – A – G – E – D – Mugas	13 + 7 + 9 + 7 + 6 = 42

Tai informatika!

Grafų teorijoje trumpiausio kelio uždavinys yra kelio radimas tarp dviejų svorinio grafo viršūnių, kad briaunų svorių suma būtų mažiausia.

Trumpiausio kelio tarp dviejų apmokestintų kelių sankirtos, kurių apmokestinimas neturi įtakos nuvažiuotam atstumui, paieška gali būti suvesta į klasikinį trumpiausio kelio uždavinį, o kelių susikirtimai vaizduojami panaudojant papildomas viršūnes ir kraštines. Klasikinis trumpiausio kelio uždavinys gali būti išspręstas naudojant Dijkstros algoritmą.

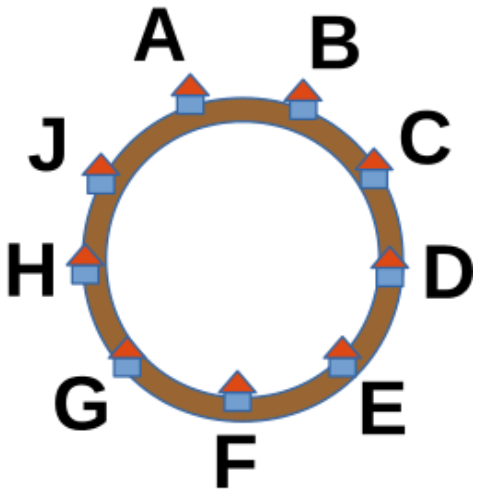
https://lt.wikipedia.org/wiki/Dijkstros_algoritmas

10. Šiurpnakčio pasivaikščiojimas

Hugo namas H yra mažame kaime, kurį sudaro 9 namai, sustatyti palei apvalų kelią aplink ežerą. Įėjimai į namus išdėstyti lygiai kas 10 metrų vienas nuo kito.

Švenčiant Šiurpnaktį Hugo užsidėjo kaukę ir išėjo lankyti kaimynų. Hugo pradėjo nuo savo namo H, pasirinko kryptį ir ėjo ta kryptimi tol, kol užsimanė aplankyti namą. Išėjęs iš namo, jis vėl pasirinko kryptį ir ėjo ta kryptimi tol, kol užsimanė aplankyti kitą namą ir t. t. Kiekvieną kartą, aplankydamas namą, Hugo užsirašydavo atstumą, nueitą nuo prieš tai aplankyto namo.

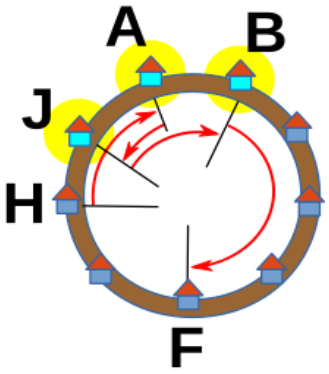
Galiausiai Hugo apsilankė name F, užrašęs šiuos skaičius: 20 10 20 40.



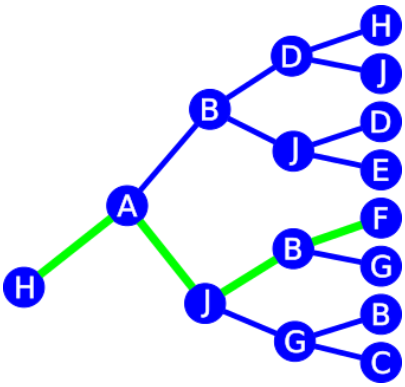
Spustelk ant trijų namų, kuriuos Hugo neabejotinai aplankė iki apsilankydamas name F.

Paaiškinimas

Hugo savo kelionės metu aplankė A, J ir B namus.

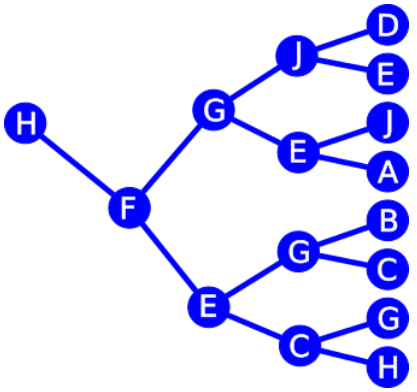


Hugo pirmiausiai pasirinko eiti pagal laikrodžio rodyklę iki A namo, tada – prieš laikrodžio rodyklę į J namą, toliau vėl pagal laikrodžio rodyklę į B namą ir galiausiai – pagal laikrodžio rodyklę į F namą.



Turime įsitikinti, kad kitokių Hugo judėjimo galimybių nėra. Atkreipiame dėmesį, kad yra 4 atstumai eiti ir kiekvienas iš šių atstumų gali būti einami pirmyn (pagal laikrodžio rodyklę) arba atgal (prieš laikrodžio rodyklę), taigi yra 2x2x2x2=16 galimų kelių. Dvi pateiktos medžių diagramos patvirtina, kad naudodamas bet kurį iš šių 16 kelių Hugo negali pasiekti F namo.

1. Jeigu Hugo pradeda pasivaikščiojimą eidamas 20 metrų pagal laikrodžio rodyklę, jis atvyksta į A namą. Pateikta medžio diagrama parodo visas galimybes einant ir pagal, ir prieš laikrodžio rodykle iš namo A.



2. Jeigu Hugo pradeda pasivaikščiojimą eidamas 20 metrų prieš laikrodžio rodyklę, jis atvyksta į F namą. Visi galimi keliai iš namo F yra parodyti antroje diagramoje.

Paskutinis aplankomas namas dešiniausiame stulpelyje abejuose grafuose ir galima matyti, kad namas F ten pasirodo tik vieną kartą, kai yra einama į A, J ir B. Taigi, tai yra vienintelis sprendimas.

Tai informatika!

Kompiuterių mokslininkai ir programuotojai dažnai turi tikro pasaulio objektus (šiuo atveju kelius, kuriais buvo lankomi kaimynai) pavaizduoti kompiuteryje kaip duomenis. Informatikai turi surasti tinkamą objekto pateikimą. Pateikimas yra geras, jeigu galime jį lengvai panaudoti tikro pasaulio objektams perkurti.

Hugo pasivaikščiojimo perkūrimas užrašant tik pradžios tašką, pabaigos tašką ir nueitus atstumus (bet ne kryptis), gali būti geras pateikimas, jeigu mums reikia žinoti visą Hugo pasivaikščiojimo atstumą (tai lengva apskaičiuoti: 20+10+20+40=90). Bet tai nėra tinkamas skaičiavimas, jeigu norime žinoti, kuriuos namus Hugo aplankė. Galite matyti, jog norint sužinoti, kuriuos namus aplankė Hugo, mums reikia išspręsti galvosūkį ir kai kuriais atvejais gali būti keletas sprendimų. Geresnis pasivaikščiojimo pateikimas šiam tikslui būtų užsirašyti tiek atstumą, tiek kryptį (panašiai kaip padaryta paaiškinime).

11. Robotas-sodininkas

Bebras nusipirko robotą mažiems medeliams sodinti. Roboto programavimo kalboje yra tokios komandos:

Pradžia: Robotas įjungiamas

Priekin(X): Robotas juda priekin X metrų

Atgal(X): Robotas juda atgal X metrų

Kairėn: Robotas pasisuka kairėn X laipsnių

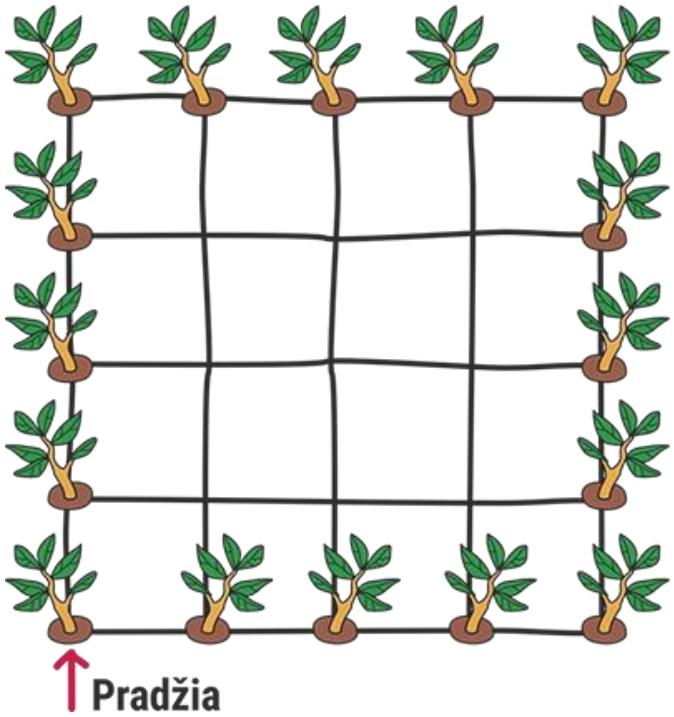
Dešinėn: Robotas pasisuka dešinėn X laipsnių

Sodink: Sodinamas mažas medelis

Kartok X{komandos}: X kartų kartojamos skliausteliuose esančios komandos

Stok: Robotas išjungiamas

Kvadrato formos sklypą galima apsodinti 16 medelių. Kiekviena sklypo kraštinė yra 8 metrų ilgio ir tarp gretimų medelių turi būti 2 metrų atstumas. Robotas pradeda darbą pradžios langelyje rodyklės kryptimi. Robotas įjungiamas komanda „Pradžia“ ir baigus darbą išjungiamas komanda „Stok“.



Kuri iš pateiktų programų tinka robotui norint apsodinti sklypą, kaip parodyta paveikslėlyje?

A.
```
Pradžia
Kartok 4{
  Kartok 4{Sodink, Priekin(2)},
  Dešinėn(90)}
Stok
```

B.
```
Pradžia
Kartok 4{
  Kartok 4{Sodink, Priekin(2)},
  Kairėn(90)}
Stok
```

C.
```
Pradžia
Kartok 4{
  Kartok 4{Priekin(1), Sodink},
  Dešinėn(90)}
Stok
```

D.
```
Pradžia
Kartok 4{
  Kartok 4{Priekin(2), Sodink},
  Kairėn(90)}
Stok
```

Paaiškinimas

Teisingas atsakymas: A.

```
Pradžia
Kartok 4{
  Kartok 4{Sodink, Priekin(2)},
  Dešinėn(90)}
Stok
```

Robotas pradeda **Kartok 4{Sodink, Priekin(2)}**, todėl jis pasodina 4 medelius sklypo viršuje. Tada jis pasisuka dešinėn, vykdydamas **Dešinėn(90)**. Ir dar 3 kartus kartoja komandas, kurios yra komandoje **Kartok 4**.

C atvejis netinka, nes komanda **Priekin**(1) neteisinga. Robotas turi nueiti 2 metrus iki kito sodinamo medelio.

B ir D netinka, nes jų programų komandos pasuka robotą kairėn ir jis išeina už sklypo ribų.

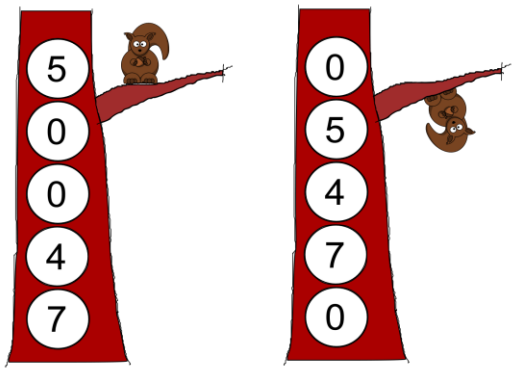
Tai informatika!

Užduotyje algoritmai siejami su jų atvaizdavimu programavimo kalba. Šiuo atveju algoritmą sudaro žingsnių seka, kurią vykdo robotas ir kuri yra suderinta su užduoties reikalavimais. Algoritmo esmė – ciklas cikle, nes reikia apeiti 4 sklypo kraštines ir einant kiekviena jų reikia atlikti tokius pat veiksmus ir ta pačia tvarka.

Pateiktose programose naudojamas ciklas cikle, t. y., vienas ciklas kito ciklo viduje. Kai turime iteraciją, tai vartojamas paprastas ciklas. Tačiau jei turime reikalų su dvimačiais objektais (šiuo atveju turime sklypą), duomenims apdoroti reikalingas ciklas cikle.

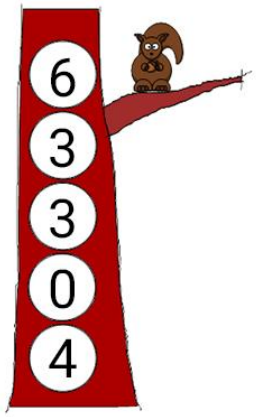
12. Savanaudės voverės

Medyje yra penkios viena virš kitos išsidėsčiusios drevės, kuriose gyvena 16 savanaudžių voverių. Kasdien kiekviena voverė apžiūri, kiek turi kaimynų drevėje, kurioje gyvena, viena dreve virš (jei yra) ir viena dreve žemiau (jei yra). Tada nustato, kurioje iš šių drevių yra mažiausias kaimynų skaičius. Naktį kiekviena voverė persikelia į atrinktą drevę (arba lieka savoje), kurioje kaimynų skaičius mažiausias. Jei tokių drevių ne viena, voverė pirmenybę teikia savo drevei, tada viršiau esančiai drevei ir galiausiai – žemiau esančiai drevei.



Šiandiena Rytdiena

Pavyzdžiui, voverės drevėse gyvena šitaip: 5, 0, 0, 4, 7 (pradedant nuo viršaus ir einant žemyn). Rytoj visos 5 viršutinio aukšto voverės apsigyvens viena dreve žemiau (0 kaimynų yra geriau nei 4). 7 voverės iš apatinės drevės persikels viena dreve aukščiau (4 kaimynai yra geriau negu 6) ir 4 voverės iš antrosios drevės nuo apačios persikels dreve aukščiau (0 kaimynų yra geriau nei 3)
Tarkime, kad voverės drevėse gyvena šitaip:



Kiek dienų reikės, kad visos voverės atsidurtų vienoje ir toje pačioje drevėje?

A. 2

B. 3

C. 4

D. Tai neįmanoma

Paaiškinimas

Teisingas atsakymas: B.

(6,3,3,0,4)->(0,9,0,7,0)->(9,0,7,0,0)->(0,16,0,0,0)

Tai informatika!

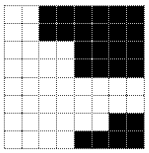
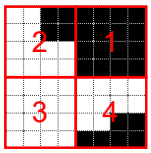
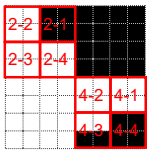
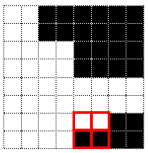
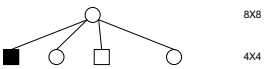
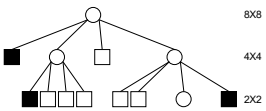
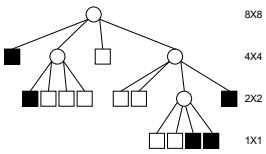
Tai skruzdžių kolonijos elgseną imituojantis uždavinys. Šio algoritmo idėja remiasi tuo, kad bet kas gali išspręsti problemas net naudodamas paprastus įrenginius, jei tų įrenginių yra labai daug. Pavyzdžiui, skruzdės juda pagal elementarias taisykles ir nepriklausomai viena nuo kitos. Tačiau kai skruzdžių daug, jos gali padaryti įspūdingus dalykus, pavyzdžiui, sulipdyti skruzdėlynus, surasti optimalų kelią grafe, išspręsti keliaujančio pirklio uždavinį ar nuskabyti lapus.

Šiame uždavinyje voverių drevių pasirinkimas pagrįstas paprastomis taisyklėmis. Tačiau čia jų elgesys toli gražu ne sumanus. Jos norėtų gyventi kuo erdviau, tačiau paprastai jos susigrūda į vieną drevę. Taigi šis uždavinys gana pamokantis: skruzdžių kolonijos algoritmą reikia taikyti apdairiai, pamatuotai, atsiminti, kad kartais geriau veikti bendradarbiaujant, o ne būti savanaudiškam.

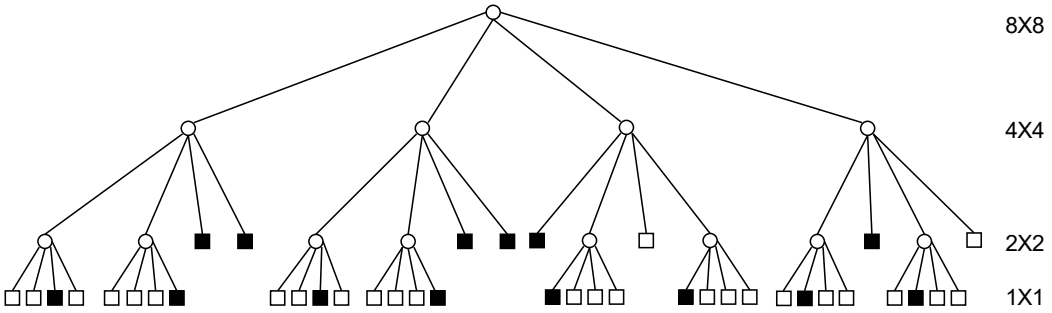
13. Lobio žemėlapiai

Bebrė piratė turi didelį lobių žemėlapį, sukarpytą į mažesnius gabalėlius. Kiekvienas žemėlapio gabalėlis yra 8 langelių pločio ir 8 langelių ilgio (1 pav.). Bebrė turi labai mažą valtį, į kurią netelpa visi žemėlapio gabalai, tad ji sugalvojo aprašyti kiekvieną regioną (žemėlapio gabalą) nedidele schema savo užrašų knygelėje. Schema sudaroma taip:

1. Jei visi langeliai regione yra tos pačios spalvos, ji nupiešia schemoje tos spalvos kvadratą.
2. Jei ne, ji nupiešia nedidelį apskritimą (1 schema) ir padalina regioną į 4 mažesnius kvadratinius regionus (2 pav.).
3. 1-as ir 2-as žingsniai kartojami tol, kol taip sužymimi visi žemėlapio langeliai (4 schema).

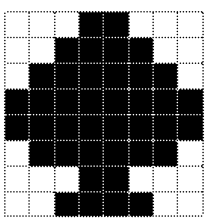
 *1 pav.*	 *2 pav.*	 *3 pav.*	 *4 pav.*
 *1 schema*	 *2 schema*	 *3 schema*	 *4 schema*

Pateikta tokia schema iš bebrės piratės užrašų knygutės:

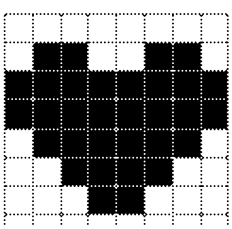


Kurį iš čia pavaizduotų žemėlapio gabalų ši schema atitinka?

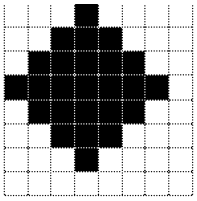
A.



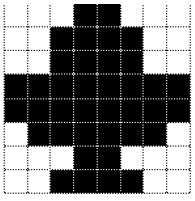
B.



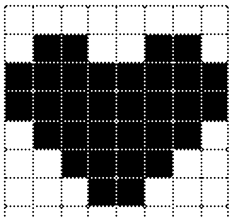
C.



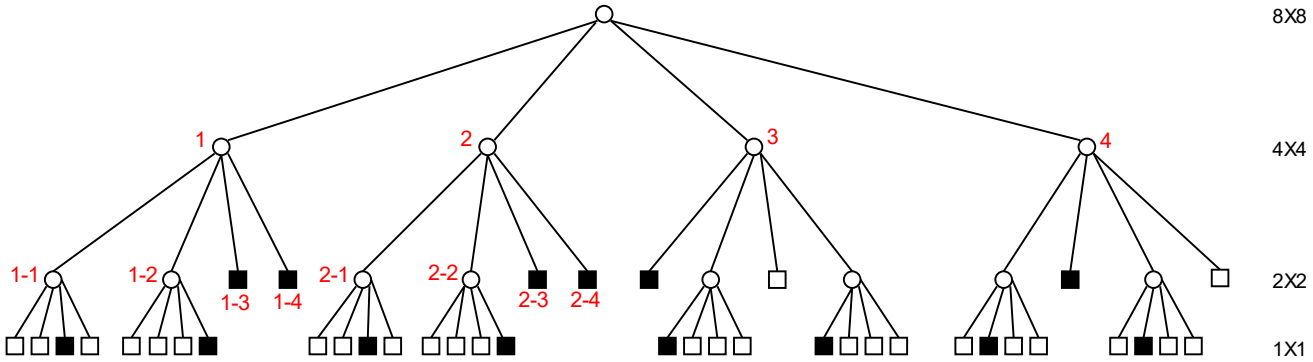
D.

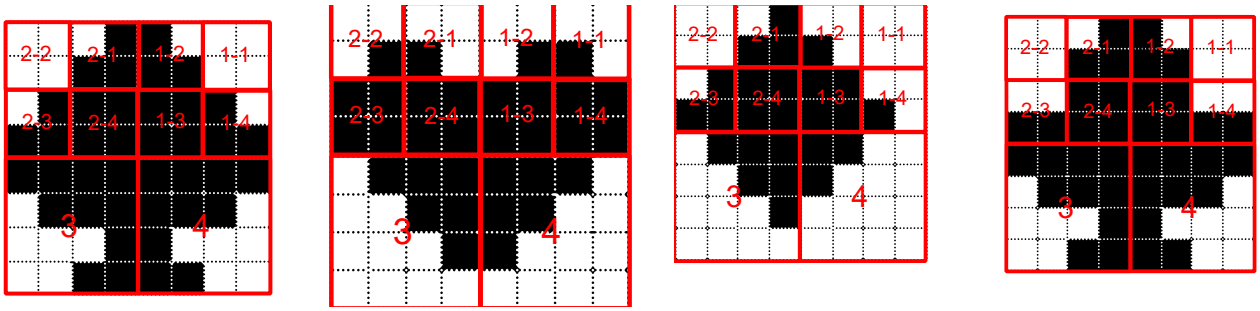


Paaiškinimas

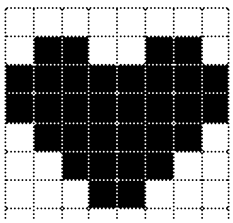
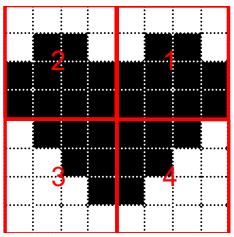
Teisingas atsakymas: 

Schemoje taškus, atitinkančius 4x4 regionus, galime sužymėti skaičiais nuo 1 iki 4, taškus, einančius iš taško 1, sužymėti nuo 1-1 iki 1-4 ir t. t. Taip surašome taškus schemoje ir juos atitinkančias vietas žemėlapiuose. 1-3, 1-4, 2-3 ir 2-4 schemoje pažymėta juodai, taigi ir žemėlapyje šie 2x2 regionai turi būti juodi – tik (B) variantas atitinka šį reikalavimą.

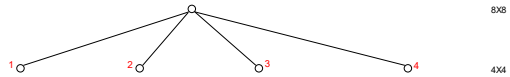




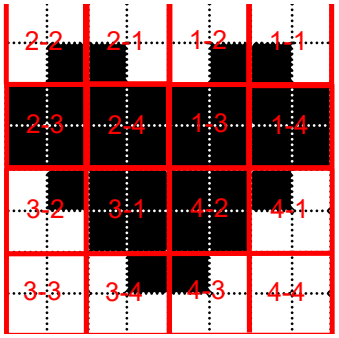
Toliau parodytas procesas, kaip žemėlapio informacija aprašoma schema. Tai parodo, kad (B) iš tikrųjų yra teisingas atsakymas.

 1 žingsnis	 2 žingsnis

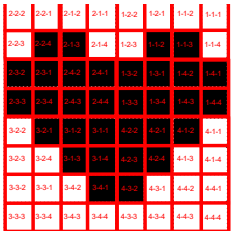
1 schema



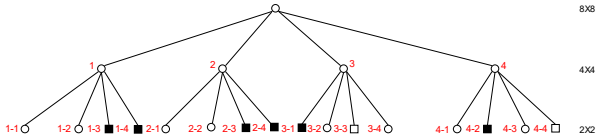
2 schema



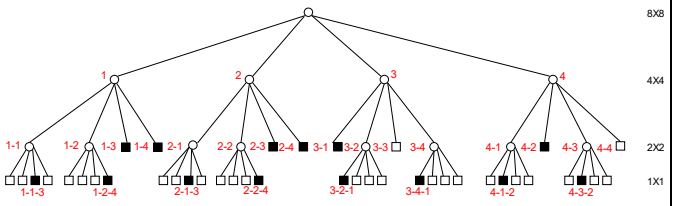
3 žingsnis



4 žingsnis



3 schema



4 schema

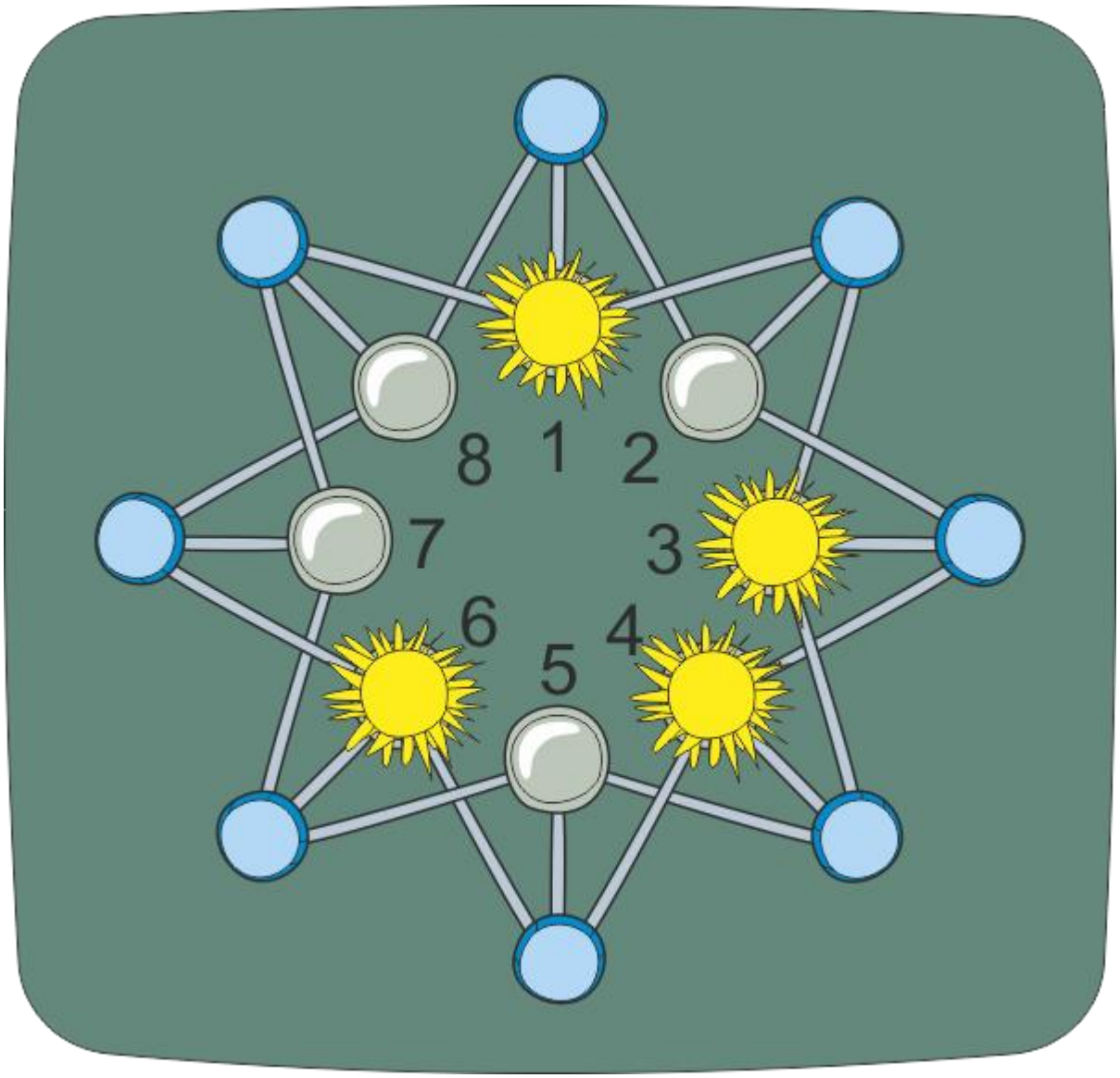
Tai informatika!

Tokia dalinimo į 4 regionus schema informatikoje vadinama ketvirtainiu medžiu (angl. *quadtree*). Ketvirtainis medis yra duomenų struktūra, kurioje kiekviena viršūnė yra arba tėvinė viršūnė lygiai 4 kitoms viršūnėms (vaikams), vaizduojamoms žemiau jos, arba yra lapas (neturi vaikų).

14. Įjunkite šviesas!

Kambaryje yra jungikliai, kuriuos spaudžiant junginėjamos lemputės. Paspaudus jungiklį, trys lemputės, kurios sujungtos su šiuo jungikliu, pakeis būsenas: jei lemputės įjungtos, jos išjungiamos, jei išjungtos – ims šviesti.

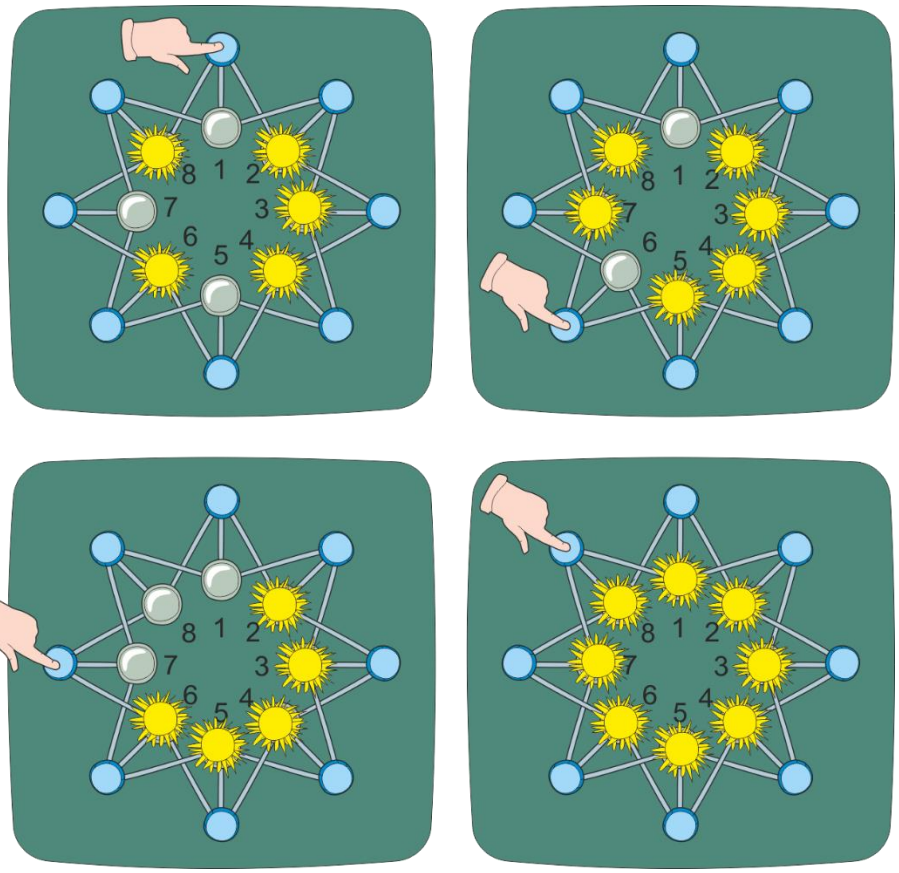
Įjunkite visas lemputes! Spaudydami po vieną jungiklį, įjunkite visas lemputes.



Paaiškinimas

Teisingas sprendimas: jungikliai, esantys šalia lempučių, kurių numeriai 1, 6, 7 ir 8, gali būti paspausti bet kuria tvarka, norint įjungti visas lemputes.

Galima pradėti nuo galutinės būsenos (tikslo) ir ieškoti sprendinio, einant pradinės būsenos link. Spustelėjus paskutinį jungiklį, jis įjungs tris lemputes tik tokiu atveju, jei trys lemputės prieš tai buvo išjungtos. Taigi pirmiausia mums reikia išjungti keletą lempučių.



Ieškant sprendinio gali pagelbėti šie pastebėjimai:

- Neverta du kartus (ar lyginį skaičių kartų) spustelėti jungiklį, nes antrasis spustelėjimas panaikina pirmojo spustelėjimo rezultatą;
- Jei spustelime jungiklį X ir po to jungiklį Y, rezultatas bus tas pats, kaip spustelėjus pirma Y, o paskui X.

Todėl galimų jungiklių spustelėjimų sekų skaičius ribotas: blogiausiu atveju visi septynių jungiklių poaibiai. Tokių poaibių 128, jei iš viso aibėje yra 7 elementai. Panaudojus kiekvieną poaibį, gausime skirtingą rezultatą. Kadangi yra tik 27 (= 128) skirtingos tinklo konfigūracijos, kiekviena jų gali būti gaunama iš pradinės konfigūracijos.

Tai informatika!

Daugelyje užduočių žinome tikslą ir pradinę būseną. Šios užduoties tikslas – įjungti visas lemputes, pradinė būsena: įjungtos 3-a, 5-a ir 7-a lemputės, o likusios lemputės išjungtos.

Kol dirbtinis intelektas dar buvo neišplėtotas, problemų sprendimas, naudojant kompiuterines sistemas, dažnai buvo suprantamas kaip veiksmų seka, vedanti nuo pradinės būsenos prie tikslo. Tokia samprata turi prasmę, jei yra baigtinė galimų veiksmų seka ir baigtinė aibė objektų, kurių būseną galima keisti, atliekant nurodytus veiksmus. Buvo sukurta daug įvairių tokią paiešką realizuojančių strategijų. Pavyzdžiui, „daugelio pabaigos būsenų analizė“ pasirenka veiksmą, kuris sumažina skirtumą tarp dabartinės ir galutinės būsenos.

Šiame uždavinyje pravartu taip pat ieškoti veiksmų, kurie pakeistų kažkurią kitą būseną galutine būsena. Jei paieška atliekama tiek iš pradinės, tiek iš galutinės būsenos, tokia paieška vadinama „dvikrypte paieška“.

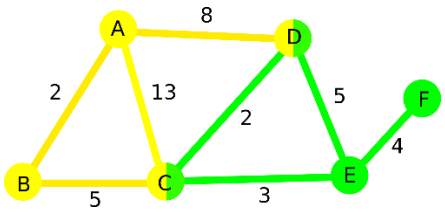
Šaltinis: https://en.wikipedia.org/wiki/Bidirectional_search

15. Ryšių tinklai

Paveikslėliuose tarptautiniai ryšių tinklai pavaizduoti skirtingomis spalvomis. Tinklus sudaro ryšių linijos ir mazgai (taškai, kuriuose susikerta viena ar kelios linijos). Dvispalviai mazgai vaizduoja skirtingų tinklų ryšius. Skaičiai virš linijų rodo siuntimo tomis linijomis kainą. Kai pranešimas siunčiamas iš vieno tinklo mazgo į kito tinklo mazgą, siuntimo kainą sudaro:

(kelio, kuriuo pranešimas siunčiamas viename tinkle, kaina) + (kelio, kuriuo pranešimas siunčiamas kitame tinkle, kaina) × 2.

Pavyzdys:

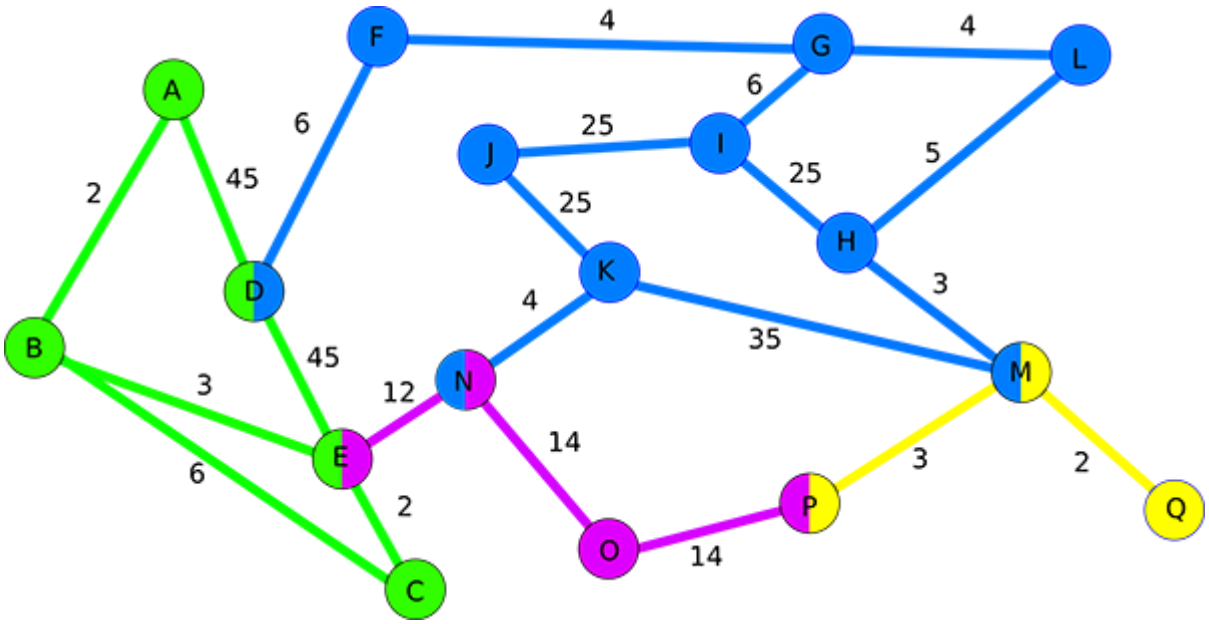


Pigiausia kaina informacijai persiųsti iš mazgo A į F yra 21:

A→B	B→C	C→E	E→F
2	5	3	4

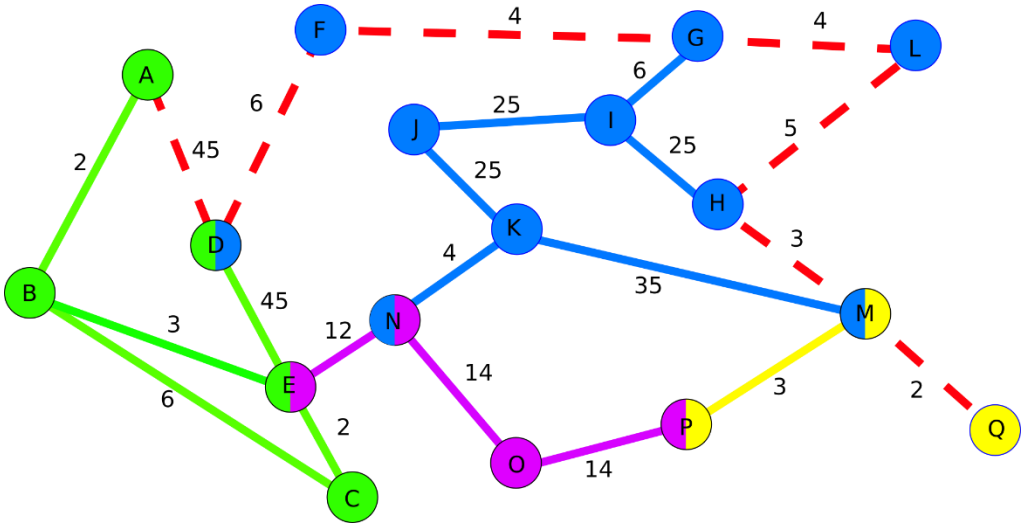
2+5+(3+4)*2=21

Kokia yra mažiausia kaina pranešimui persiųsti iš mazgo A į Q šiais tinklais?



Paaiškinimas

Teisingas atsakymas: 93.



Atkreipsime dėmesį, kad pigiausias kelias ne visada yra trumpiausias. Kad pranešimas iš žaliojo tinklo nukeliautų iki violetinio tinklo, mažiausia kaina 5. Tačiau iš mazgo E į Q keliaujant mazgais N, K, M arba mazgais N, O, P, M, kaina didesnė, negu keliaujant mazgais D, F, G, L, H, M.

Tada:

- Kelio A, D, F, G, L, H, M, Q kaina būtų 45 + (6 + 4 + 4 + 5 + 3 + 2) × 2 = 93
- Kelio A, B, E, N, K, M, Q kaina būtų (2 + 3) + (12 + 4 + 35 + 2) × 2 = 111
- Kelio A, B, E, N, O, P, M, Q kaina būtų (2 + 3) + (12 + 14 + 14 + 3 + 2) × 2 = 95

Tai informatika!

Šiame uždavinyje reikia rasti mažiausios kainos kelią grafe iš vieno mazgo į kitą, kreipiant dėmesį į pografių sujungimo taškus. Kelio grafe radimas – vienas svarbesnių informatikos uždavinių, taikomas daugelyje sričių.

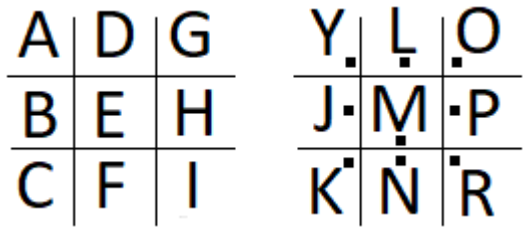
Pigiausiam keliui rasti galima naudoti, pavyzdžiui, Dijkstros algoritmą. Šis algoritmas taikomas daugelyje informatikos sričių, kai ieškoma trumpiausio kelio, pavyzdžiui., kompiuterių tinkluose, maršrutų parinkimo sistemose. Šį algoritmą naudoja „Google“ žemėlapiai trumpiausiam maršrutui rasti iš vienos geografinės vietos į kitą. Biologijoje šis algoritmas taikomas konstruojant infekcinių ligų plitimo modelį. Deja, dėl šiame uždavinyje pasirinkto būdo kainoms skaičiuoti Dijkstros algoritmas negali būti pritaikytas tiesiogiai.

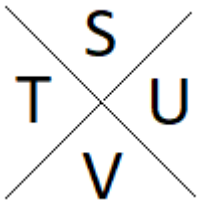
https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

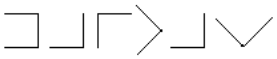
16. Masonų šriftas

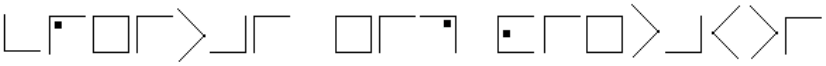
Jurgis ir jo kaimynė Agota bendrauja slaptais pranešimais, kuriuos perduoda vienas kitam pro sodo tvorą. Jie sukūrė raktus, kad niekas negalėtų perskaityti šių pranešimų.

Naudojami raktai:





Pavyzdžiui, žodis BAITAS užkoduojamas taip: 

Jurgis gavo Agotos pranešimą: 

Kokį pranešimą gavo Jurgis?

A. „Greitai eik pietauti“

B. „Skubiai eik dainuoti“

C. „Skubiai eik skaityti“

D. „Greitai eik skaityti“

Paaiškinimas

Jurgis gavo pranešimą „Greitai eik pietauti“. Galima pradėti nuo pirmo žodžio iššifravimo. Išsiaiškinę, kad pirmas žodis yra „greitai“, toliau nustatome paskutinio žodžio pirmąją raidę. Randame, kad tai raidė „p“, taigi ir žodis bus „pietauti“.

Tai informatika!

Šis kodavimas yra keitinių šifras – kiekviena raidė pakeičiama atitinkamu simboliu. Be tinklelių schemos (šifro rakto) iššifruoti tekstą būtų sunkiau (geresnių šifrų iššifruoti beveik neįmanoma). Informatikoje šifravimas yra svarbus saugumui užtikrinti.

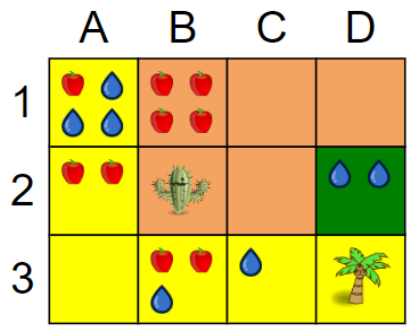
Koduojama dėl įvairių priežasčių: patogesnio informacijos vartojimo, techninių galimybių, informacijos apsaugos (kad neperskaitytų pašaliniai). Kai kalbama apie duomenų apsaugą, tikslingiau vartoti šifravimo sąvokas (šifras, šifravimas ir pan.).

Daugiau skaitykite: https://en.wikipedia.org/wiki/Pigpen_cipher

17. Dykuma

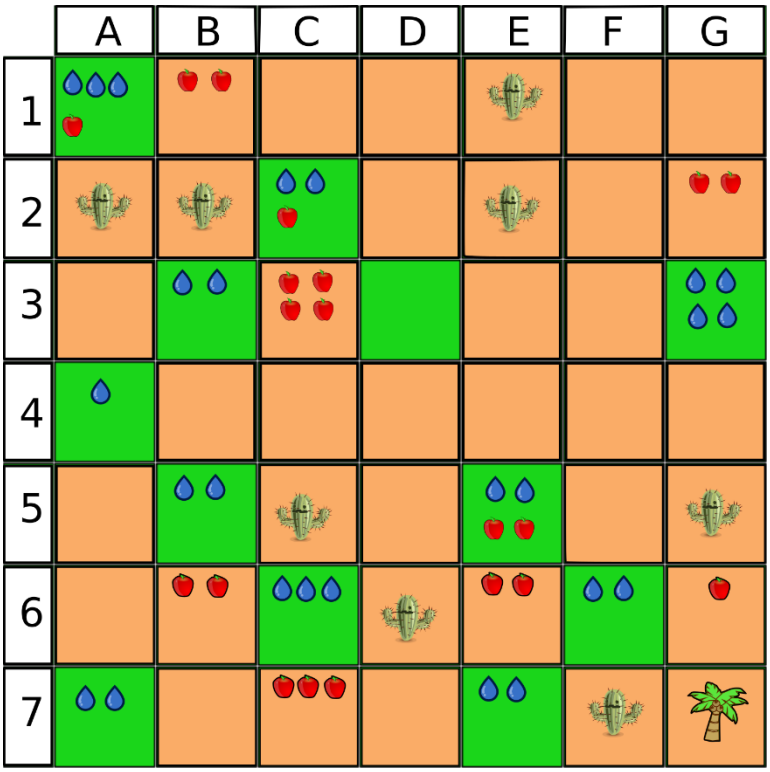
Lukas keliauja aplink pasaulį. Aplankęs daug vaizdingų vietovių, pasiekė dykumą. Kelionę per dykumą pradeda langelyje A1 ir turi pasiekti palmę langelyje D3. Lukas gali eiti kairėn, dešinėn, aukštyn ir žemyn, bet negali eiti įstrižai. Per tą patį langelį gali eiti tik kartą. Taip pat turi vengti langelių su kaktusu.

Lukas turi rinkti langeliuose esančius obuolius ir vandens lašus. Pereinant iš vieno langelio į kitą netenkama vieno lašo vandens ir vieno obuolio. Jei Lukas neturi vandens ir obuolio, nebegali toliau keliauti. Pavyzdyje geltona spalva pažymėtas teisingas dykumos perėjimo kelias:



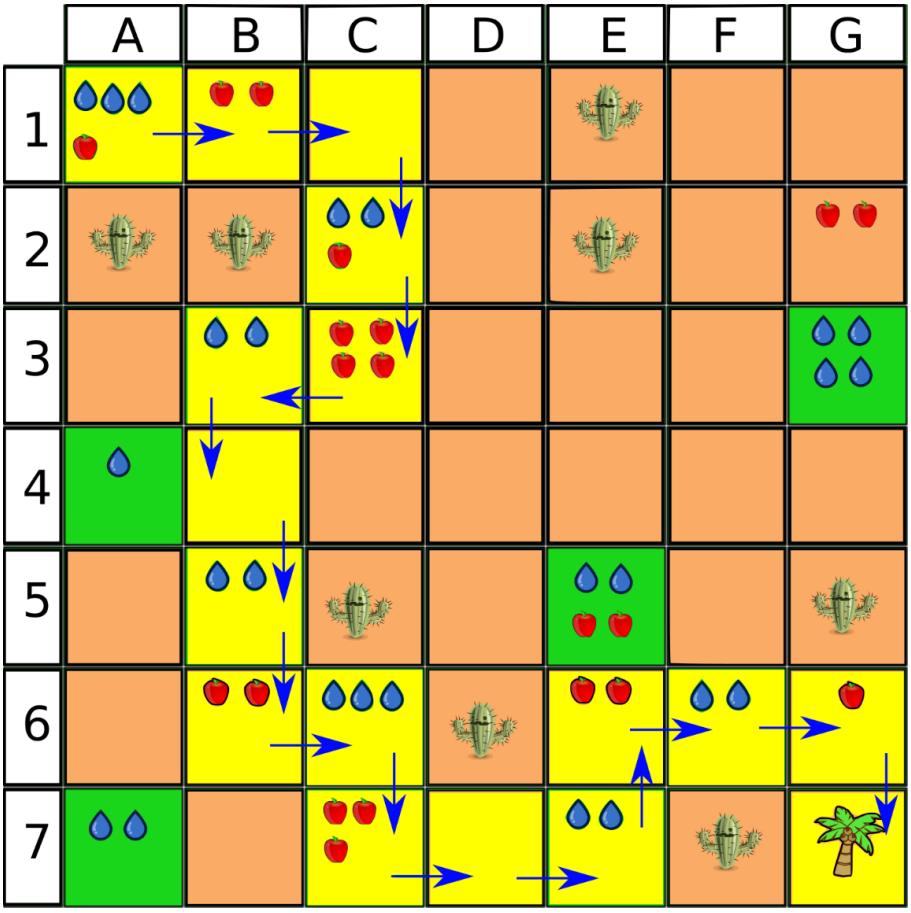
- A1: turi 3 lašus ir 1 obuolį
- A2: liko 2 lašai ir 2 obuoliai
- A3: liko 1 lašas ir 1 obuolys
- B3: liko 1 lašas ir 2 obuoliai
- C3: liko 1 lašas ir 1 obuolys
- D3: tikslas pasiektas!

Padėkite Lukui pereiti dykumą ir pasiekti langelį su palme.



Paaiškinimas

Teisingas atsakymas:



Yra du galimi keliai: vienas pavaizduotas paveiksle, kitas toks pat, tik dar aplankant E5 ir E6. Visuose kituose keliuose pritrūktų vandens arba obuolių.

Tai informatika!

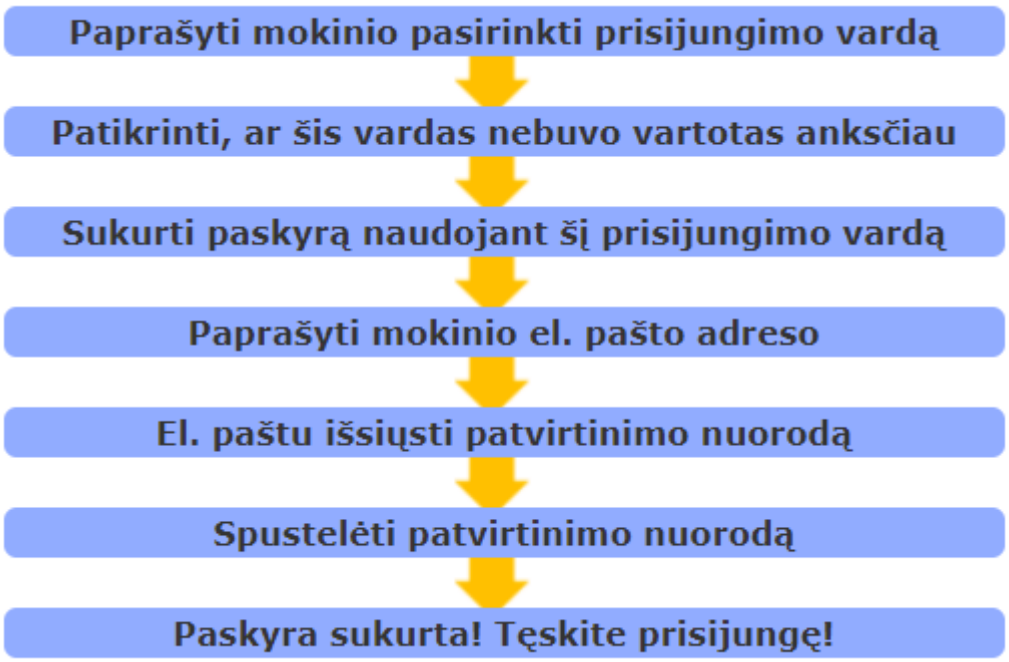
Užduotis remiasi gerai žinomu informatikos mokslo paieškos į gylį algoritmu.

Paieška į gylį (angl. *depth-first search*, DFS) – paieškos grafe arba grafo apėjimo būdas, kai pasirinkus pradinę viršūnę einama grafo briaunomis kiek įmanoma giliau, renkantis vis naują viršūnę; kai paskutinė aplankyta viršūnė naujos (dar neaplankytos) kaimynės nebeturi, tada grįžtama iki artimiausios neaplankytos briaunos ir vėl ieškoma kuo giliau tol, kol bus rastas ieškomas tikslas arba kol bus aplankytos visos grafo viršūnės ir briaunos.

Šis uždavinys reikalauja gebėjimo atlikti kelias užduotis vienu metu. Daugelio užduočių atlikimas vienu metu yra tiek žmogaus gebėjimas, tiek kompiuterio funkcija. Tokio sprendimo ieškojimas reikalauja vienu metu apdoroti kelis susijusius kintamuosius.

18. Paskyros kūrimas

Aušra kuria klasės svetainę, kad klasės draugai galėtų dalytis meno projektais. Ji projektuoja procesą vartotojo paskyrai susikurti, kai naudojamas unikalus vartotojo vardas ir el. pašto adresas. Proceso žingsniai vaizduojami schema:

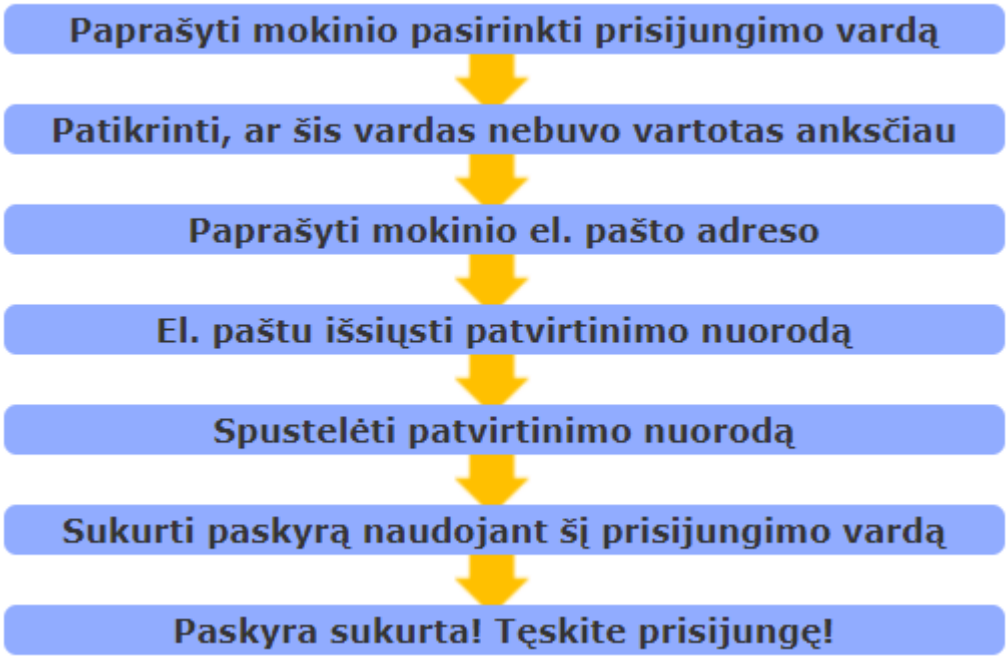


Testuodama projektuojamą procesą Aušra surado svarbią klaidą. Jei vartotojas klaidingai nurodo el. pašto adresą, jis negauna sukurtą paskyrą patvirtinančio laiško ir negali prisijungti. Vartotojui nebeleidžiama jungtis iš naujo, nes jo sukurtas vardas jau buvo patvirtintas.

Padėkite Aušrai sutvarkyti prisijungimo procesą. Perdėliokite proceso žingsnius taip, kad klaida būtų ištaisyta.

Paaiškinimas

Paskyra turi būti kuriama tiktai tada, kai gaunamas patvirtinimas (atpažįstamas, kai spustelima nuoroda, kuri buvo išsiųsta pateiktu el. pašto adresu). Taigi trečiasis žingsnis turi būti šeštuoju – nukeltas į priešpaskutinę poziciją.



Tai informatika!

Šis uždavinys skirtas supažindinti su sistemų projektavimu – informacinių sistemų kūrimu procesu siekiant tenkinti specifinius vartotojo reikalavimus. Vienas iš reikalavimų – sistemos projektavimo loginė kontrolė. Sistema turi tenkinti vartotojo poreikius, atitikti reikalavimus, taip pat automatiškai atlikti savo pačios užduotis, kurios apiforminamos programa, taigi turi valdyti programos darbą.

Srauto valdymo analizė padeda derinti projektuojamą sistemą – surasti klaidas ar trūkumus, spręsti technines problemas.

Nagrinėjant išsamiau, šiame uždavinyje aprašytas procesas iš tiesų turi dvi užrakinimo fazes. Pirmoji užrakinimo fazė nutinka, kai pasirenkamas vardas (kai reikia identifikuoti mokinį). Antroji fazė turėtų panaikinti užstrigimą, kai gaunamas patvirtinimas iš sistemos. Tačiau pagal pateiktą sąlygoje schemą, tai neįvyksta. Pataisius schemą, išvengiama šios problemos.

Pastebėkime, kad lygiagrečiose aplinkose šis sprendimas neišspręstų problemos, kadangi du mokiniai vienu metu gali pateikti tą patį vardą.

Atkreipiame dėmesį, kad pateikta schema nėra struktūrinė diagrama, kuri nurodytų visus galimus veiksmus ir sprendimus. Šia diagrama tik norima pavaizduoti projektavimo procesą, bet ne pateikti visą paskyros kūrimo procedūrą.

19. Žemėlapio žaidimas

Bebras prisiregistravo žaisti žemėlapio žaidimą. Žaidžiant visi keliai laikomi vienos krypties. Žaidėjams nebuvo duotas joks žemėlapis, jie gavo tik tokią lentelę:

Yra šeši miestai: A, B, C, D, E ir F. Rodyklė lentelėje reiškia vienos krypties eismo kelią, prasidedantį pateiktos rodyklės eilutės mieste ir pasibaigiantį jos stulpelio mieste. Kai rodyklės nėra, reiškia, joks kelias nejungia tų dviejų miestų. Pavyzdžiui, yra vienos krypties eismo kelias iš miesto B į C, bet nėra jokio vienos krypties eismo kelio iš miesto B į D.

Kuris iš pateiktų teiginių yra teisingas?

A. Bebrui reikia panaudoti mažiausiai tris vienos krypties eismo kelius, kad nuvyktų iš miesto C į miestą B.

B. Bebrui reikia panaudoti mažiausiai tris vienos krypties eismo kelius, kad nuvyktų iš miesto A į miestą D.

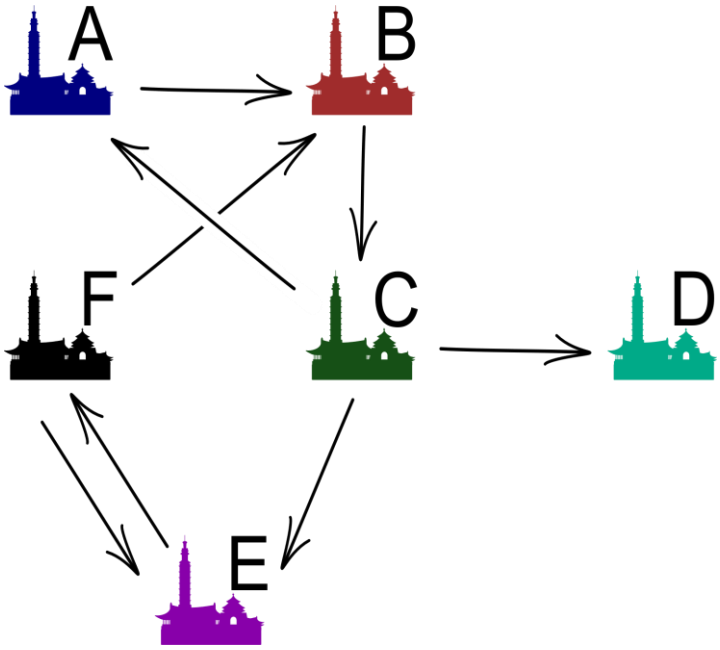
C. Bebrui reikia panaudoti mažiausiai tris vienos krypties eismo kelius, kad nuvyktų iš miesto A į miestą C.

D. Bebrui reikia panaudoti mažiausiai tris vienos krypties eismo kelius, kad nuvyktų iš miesto B į miestą E.

Paaiškinimas

Teisingas teiginys: B.

Bebrui reikia panaudoti mažiausiai tris vienos krypties eismo kelius, kad nuvyktų iš miesto A į miestą D.



Paveikslėlis vaizduoja vienos krypties eismo kelius tarp miestų.

(A) atveju iš miesto C keliaujama į miestą A, toliau – į miestą B naudojantis iš viso dviem keliais. Todėl atsakymas A yra neteisingas.

(B) atveju iš miesto A keliaujama tik į miestą B, toliau – į miestą C, ir galiausiai – į miestą D. Šis atsakymas yra teisingas.

(C) atveju iš miesto A keliaujama į miestą B, toliau – į miestą C naudojantis iš viso dviem keliais. Todėl atsakymas D yra neteisingas.

(D) atveju iš miesto B keliaujama į miestą C, toliau – į miestą E naudojantis iš viso dviem keliais. Todėl atsakymas D yra neteisingas.

Tai informatika!

Sprendime pateiktas žemėlapis yra tiesioginis grafas (dar vadinamas digrafu). Tiesioginis grafas susidaro iš viršūnių (miestai) ir jas jungiančių tiesioginių kraštinių (vieno eismo keliai).

Yra daugybė tiesioginio grafo vaizdavimo būdų. Pavyzdžiui, galime naudoti gretimumo matricą, panašią į uždavinyje pateiktą lentelę.

Duomenų vaizdavimas įvairiais būdais labai populiarus kompiuterijoje. Svarbu suprasti kiekvieno vaizdavimo būdo pranašumus ir trūkumus. Pavyzdžiui, paveikslėliai yra vaizdūs, kai juos nagrinėja žmogus, tačiau kai duomenis apdoroja kompiuteriai, geriau juos vaizduoti matematinėmis lentelėmis.

20. „Soundex“ algoritmas

„Soundex“ algoritmas sukurtas vardams atpažinti nepaisant nedidelių jų pateikimo skirtumų. Iš esmės jis skirtas anglų kalba tariamiems vardams indeksuoti.

Kiekvienam vardui generuojamas 4 skaitmenų kodas pagal šias taisykles:

1. Palikti pirmąją žodžio raidę.

2. Pašalinti visas šias raides: A, E, I, O, U, H, W, Y.

3. Likusias raides pakeisti skaitmenimis iš lentelės:

Raidės	*Skaitmenys*
B, F, P, V	1
C, G, J, K, Q, S, X, Z	2
D, T	3
L	4
M, N	5
R	6

4. Du ar daugiau iš eilės einančius tuos pačius skaitmenis pakeisti vienu tuo pačiu skaitmeniu.

5. Imti tik pirmuosius keturis žodžio simbolius; jei žodis trumpesnis, pabaigoje pridėti nulių.

Pavyzdžiai:

Žodis	Kodas
BEBRAS	B162
VAIDOTAS	V320
LINA	L500
VALENTINA	V453

Koks kodas būtų sugeneruotas vardui ZIGMANTAS?

A. Z255

B. Z250

C. Z253

D. Z252

Paaiškinimas

Teisingas atsakymas: Z253.

1. Paliekama pirmoji žodžio raidė (Z).

2. Zigmantas → Z25532 (pritaikius pirmąsias tris taisykles).

3. Z2532 (iš eilės einantys skaitmenys pakeičiami į vieną.)

4. Z253 (paliekami tik pirmieji keturi simboliai).

Tai informatika!

„Soundex“ yra fonetinis algoritmas, skirtas indeksuoti vardus pagal garsus, tariamus anglų kalba. Siekiama, kad homofonai (vienodai tariami, bet skirtingai rašomi žodžiai) būtų užkoduoti vienodai, kad juos būtų galima sugretinti nepaisant nedidelių rašybos skirtumų.

„Soundex“ algoritmas daugiausia koduoja priebalsius; balsis nebus užkoduotas, nebent tai būtų pirmoji raidė. Šis algoritmas yra standartinė duomenų bazių programinės įrangos funkcija.

„Soundex“ algoritmas taikomas fonetinių klaidų korekcijai: klaidoms, atsirandančioms rašant vardų sekas, ypač kai yra panašiai tariamų žodžių. Algoritmą naudoja įvairios tarptautinės institucijos (pavyzdžiui, Interpolas), tačiau kai kurių šalių vardams jis netinka.

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

Algoritmas išpopuliarėjo, kai Donaldas Knuthas jį aprašė savo knygoje „The Art Of Computer Programming, vol. 3: Sorting and Searching“.

21. Sesės ir brolio paslaptys

Lina ir jos brolis Aras mėgsta rašyti žinutes, kurios užšifruojamos jų sukurta kodavimo sistema. Koduojamos tik raidės, kiti ženklai nekoduojami.

Vieną kartą Aras parašė sesei tokią žinutę:

AS ŠCFDM SĄSOI? – ICYE

Aras žinutes visada užbaigia parašydamas savo vardą.

Naudodami tą pačią kodavimo sistemą, užkoduokite žodį ZYLĖ ir jo kodą įrašykite į laukelį:

Lietuviška abėcėlė:

A Ą B C Č D E Ę Ė F G H I Į Y J K L M N O P R S Š T U Ų Ū V Z Ž

Paaiškinimas

Iššifruojant koduotus pranešimus labai svarbu atkreipti dėmesį į jų struktūrą ir užuominas. Dažnai naudojamos įvairios spėlionės. Pavyzdžiui, Antrojo pasaulinio karo metu bandant iššifruoti „Enigmos“ pranešimus buvo atkreiptas dėmesys, kad oro prognozė buvo skelbiama kiekvieną dieną tuo pačiu metu, todėl buvo prasmės ieškoti žodžio „oras“.

Aro žinutės atveju iš karto aišku, kad raktas gali būti surandamas naudojant jo vardą. Tolesnis nagrinėjimas rodo, kad raidės yra perkeliamos tokia tvarka:

A + 12, keičiama į I

R + 13, keičiama į C

A + 14, keičiama į Y

S + 15, keičiama į E

Nagrinėjant žinutės pradžią paaiškėja, kad pirma raidė nekeičiama, o toliau kiekviena raidė keičiama tolimesne raide nuosekliai padidinant perkėlimo skaičių vienetu. Todėl tikra žinutė yra:

AR RADAI KNYGĄ? – ARAS

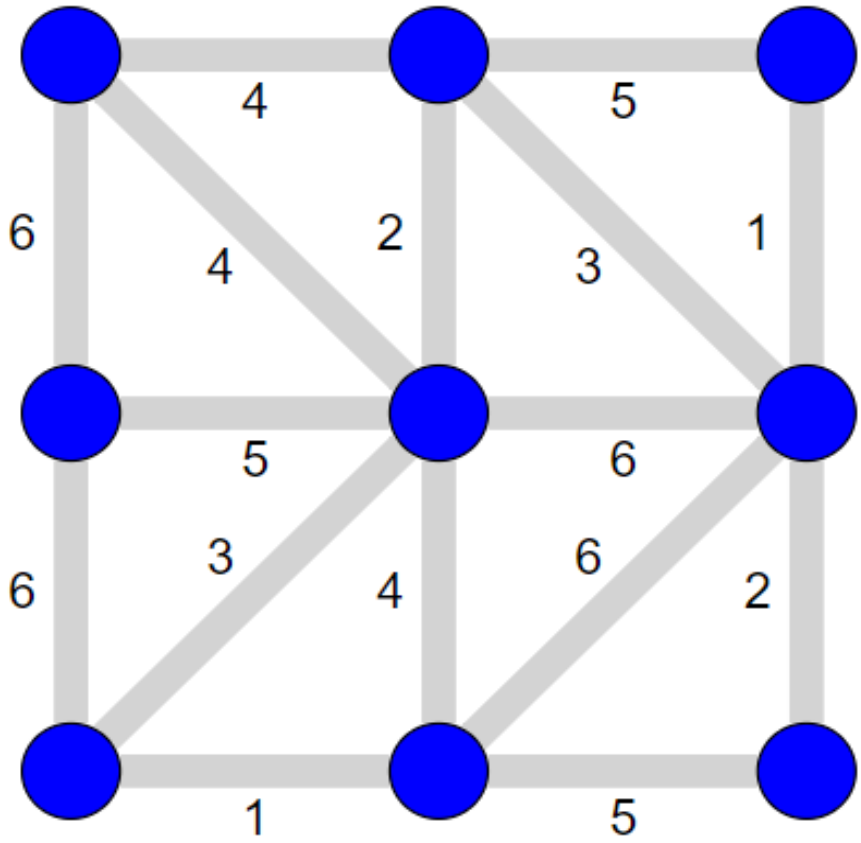
Tai informatika!

Duomenų šifravimas ir iššifravimas yra svarbi informatikos sritis.

22. Šviesolaidžių tinklas

Interneto paslaugų tiekėjas nori sukurti naują tinklą. Devyni miestai turėtų būti sujungti taip, kad kiekvienas miestas galėtų siųsti ir gauti pranešimus vienas iš kito.

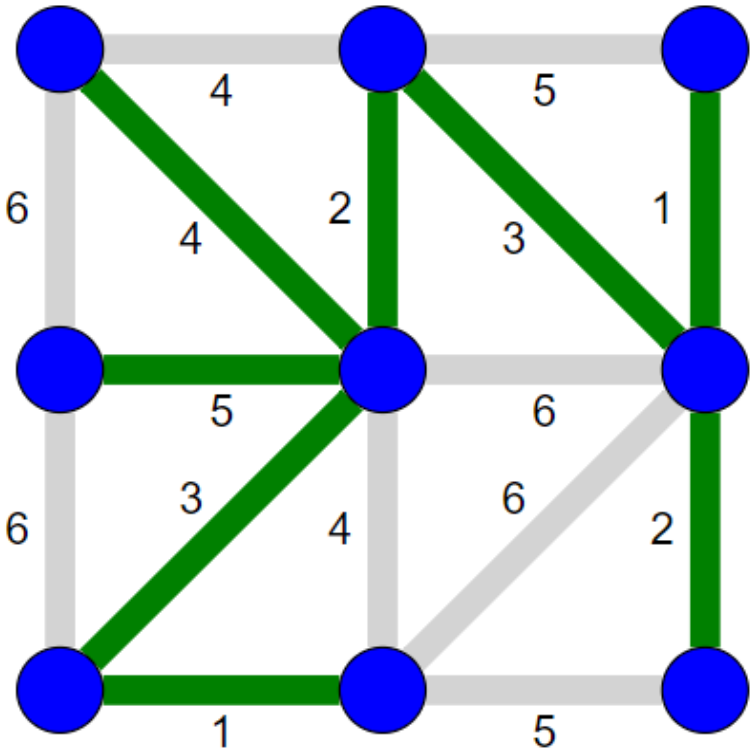
Įmonė, norėdama sukurti ryšių linijas tarp miestų, turi sumokėti tam tikrą pinigų sumą. Kainos nurodytos ant linijų, jungiančių miestus.



Kuriuos ryšius įmonė turėtų sukurti, kad kaina būtų mažiausia?

Paaiškinimas

Teisingas atsakymas:



Tai informatika!

Turime rasti būdą, kaip mažiausiomis sąnaudomis sujungti visus miestus. Ryšius tarp miestų galime pavaizduoti grafu, kuriame kiekvienas miestas yra viršūnė, o kiekviena jungtis – grafo briauna. Turime rasti tokį briaunų aibės poaibį, kuris sudarytų medį, kiekvienas miestas turėtų kelią į bet kurį kitą miestą. Turime rasti minimalų jungiamąjį medį.

Tai optimizavimo uždavinys, kurį sprendžia Primo algoritmas – godusis algoritmas, kuriuo randamas minimalus jungiamasis medis. Primo algoritmu minimalus jungiamasis medis konstruojamas prijungiant grafo briaunas: pradedama nuo medžio, kurį sudaro viena laisvai pasirinkta viršūnė, prijungiamoji briauna turi būti pigiausia ir lygiai viena briaunos viršūnė turi priklausyti konstruojamam medžiui (pastaroji sąlyga garantuoja, kad prijungiant briauną nesusidarys ciklas).

Algoritmą 1930 m. atrado čekų matematikas Vojtechas Jarnikas (Vojtěch Jarník), o vėliau 1957 m. jį iš naujo sukūrė ir paskelbė Robertas Klėjus Primas (Robert Clay Prim).

Algoritmą teoriškai galima apibūdinti kaip atliekantį šiuos veiksmus:

1. Inicializuojamas medis su viena viršūne, pasirinkta iš grafo.

2. Medis padidinamas viena briauna: iš briaunų, jungiančių medį su viršūnėmis, kurių dar nėra medyje, surandama mažiausio svorio briauna ir ji prijungiama į medį.

3. Kartojamas 2-as veiksmas (kol visos viršūnės bus medyje).

Daugiau informacijos: https://inf-knyga.nmakademija.lt/lt/latest/11_med%C5%BEiai.html

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.