ISSN 2783-7599

b·e·b·r·a·s

Informatikos ir informatinio mąstymo uždavinių rinkinys

Nr. 10

ŠVIETIMO, MOKSLO IR SPORTO MINISTERIJA

VILNIAUS UNIVERSITETAS · 1579 · UNIVERSITAS VILNENSIS

Bebras
International Challenge on Informatics and Computational Thinking

Šiame rinkinyje pateikiami 2022 metų XVIII informatikos ir informatinio mąstymo konkurso (iššūkio) „Bebras“ II 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 9–12 klasių mokinių informatikos ir 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, gilina jų supratimą, ugdosi gebėjimą taikyti jas praktiškai, uždavinių sprendimus aptaria su bendraamžiais ir mokytojais.

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

Dėkojame Daumilui Ardickui, Nojui Gudinavičiui, dr. Tatjanai Jevsikovai, Audronei Klupšaitei, Linui Kurtinaičiui, Alvidai Lozdienei, Aleksandrui Maliuginui, Vaidai Masiulionytei-Dagienei, Skirmantei Matažinskaitei, Algirdui Navickui, Jonui Ruigiui, dr. Gabrielei Stupurienei, Tomui Šiauliui, Gvidui Tupiniui, talkinusiems verčiant ir adaptuojant uždavinius. Taip pat dėkojame tarptautinei „Bebro“ bendruomenei ir uždavinių autoriams.

Parengė Lina Vinikienė

Konsultavo Valentina Dagienė

Redagavo Viktoras Dagys

Viršelį kūrė Vaidotas Kinčius

Įvadas

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

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

Mokiniai, spręsdami įdomius, informatikos konceptais grįstus uždavinius, susipažįsta su informatikos sąvokomis, pagilina jų supratimą, išsiugdo gebėjimą taikyti jas praktiškai, uždavinių sprendimus aptaria su bendraamžiais ir mokytojais. Didelė dalis uždavinių yra iš algoritmų ir programavimo srities. Juos perpratus, skatinama mokytis praktinio programavimo. Mokytojams uždaviniai – didaktiniai ištekliai, idėjos savo pamokoms paįvairinti. Mokslininkai, informatikos tyrėjai, kurdami uždavinius stengiasi perteikti esminius informatikos mokslo principus. „Bebro“ bendruomenę labiausiai vienija patrauklių, įdomių informatikos uždavinių paieška, informatikos konceptų išreiškimas žaidybinėmis užduotimis.

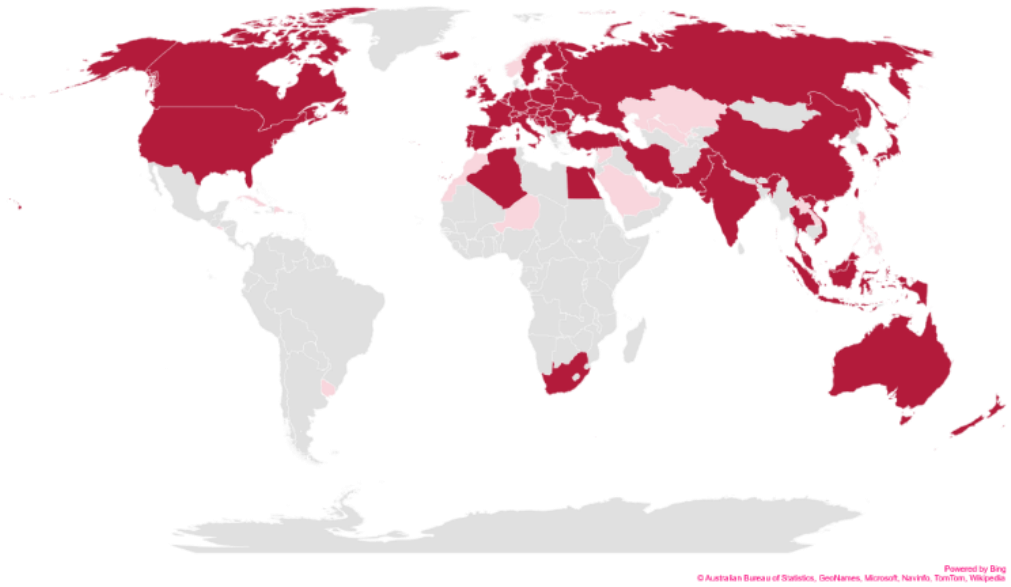
Vykdant konkursą kaupiami mokinių sprendimai – daugybė duomenų surinkta, galima atlikti tyrimus, stebėti ir panašiai. Įvairių šalių mokslininkai, remdamiesi „Bebro“ sukauptais duomenimis, rengia ir publikuoja mokslinius straipsnius

(žr. https://www.bebras.org/publications.html).

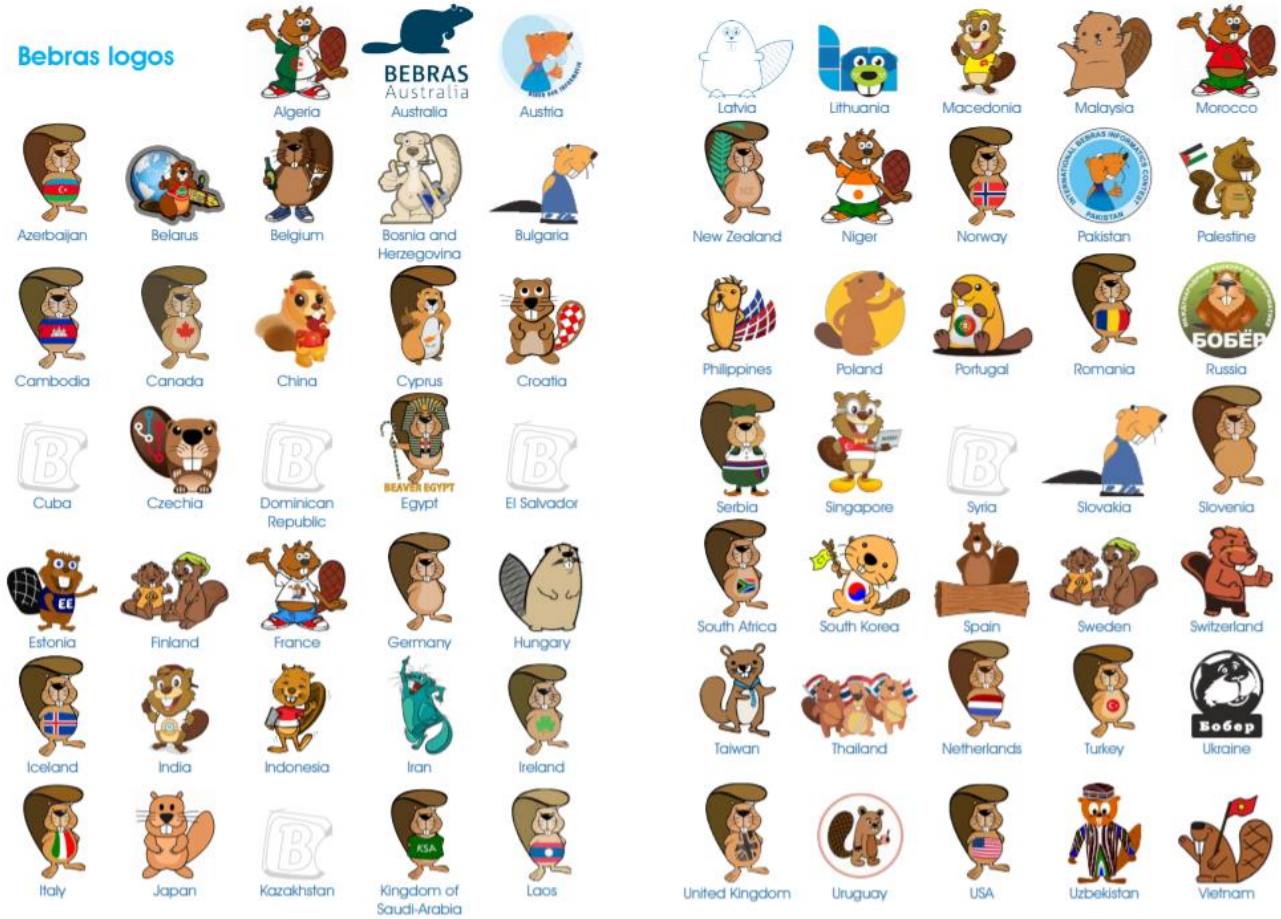
2021 metų „Bebro“ uždavinius sprendė 3 033 518 mokinių 57-iose valstybėse. Lietuvoje tais metais konkurse dalyvavo 50 638 mokiniai.

Daugiau informacijos apie „Bebro“ konkursą pateikiama interneto svetainėse:

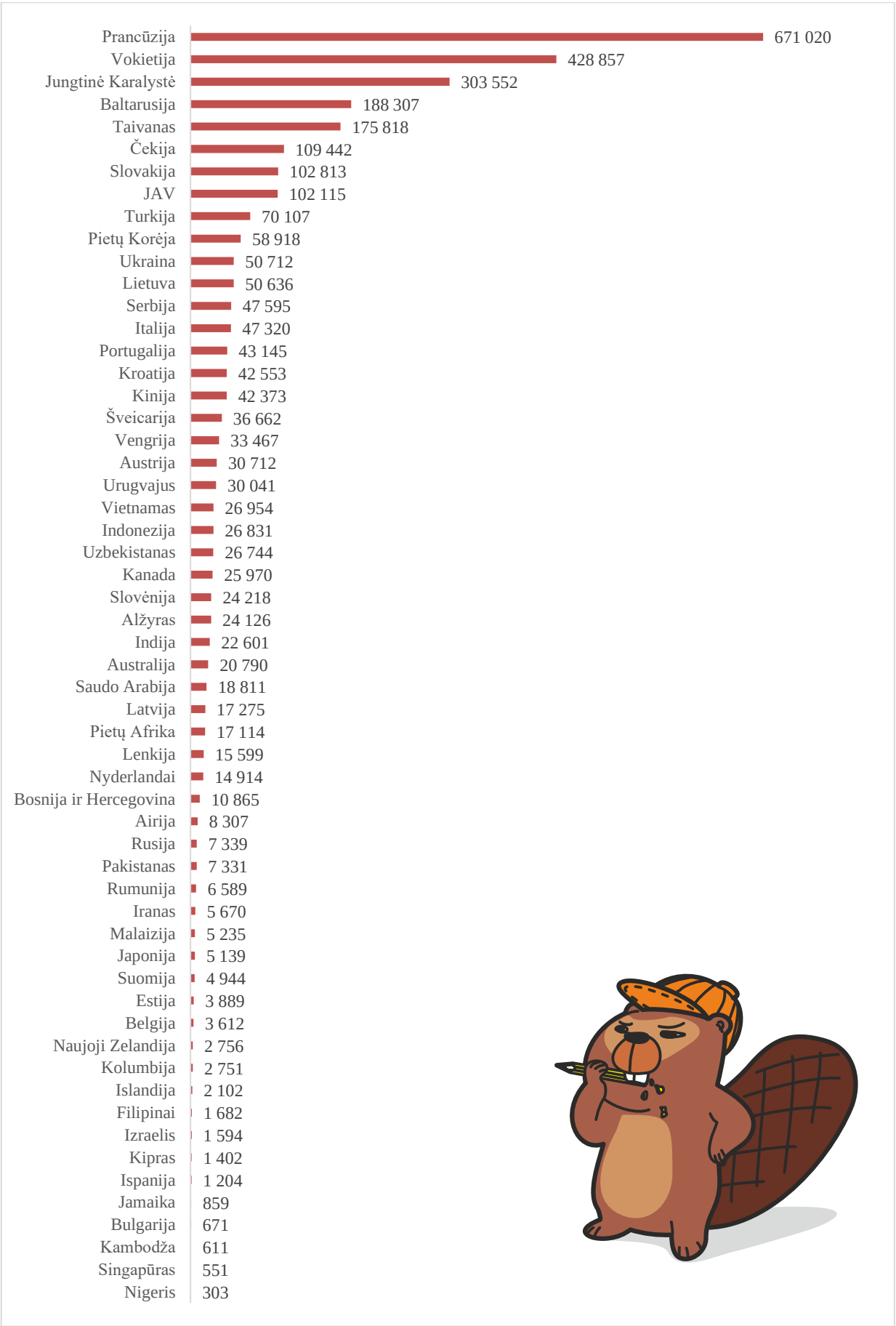
- Tarptautinė „Bebro“ iššūkio svetainė: www.bebras.org
- Lietuvos „Bebro“ svetainė: bebras.lt
- Konkurso sistema – vadinamasis „Bebro“ varžybų laukas: lt.bebras.lt



„Bebro“ pasaulyje – 51 šalis narė, 16 šalių kandidačių



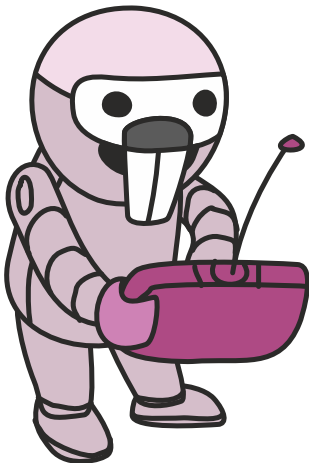
„Bebro“ konkurso šalių logotipai



2021 m. „Bebro“ dalyvių skaičius pagal šalis (57 šalių duomenys)

Kai kurios šalys geriausiems dalyviams rengia akivaizdinį antrąjį etapą. Lietuvoje antrasis etapas rengiamas universitetuose ir kolegijose sausio mėnesio pabaigoje ar vasario pradžioje.

2022 m. vasario 5 d. Lietuvoje II etape jauniai ir kolegos sprendė po 15 uždavinių: trečdalis buvo lengvesnių, trečdalis – vidutinio sunkumo ir trečdalis – sunkesnių. Uždaviniams spręsti skiriama 30 minučių.



Lietuvoje konkurso taškai skaičiuojami taip:

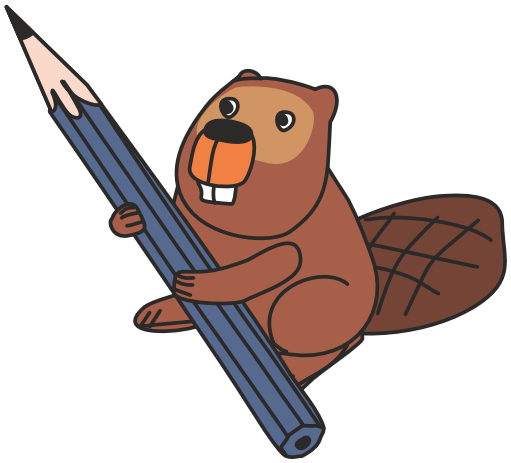
- Prieš pradėdamas spręsti, kiekvienas dalyvis turi 45 taškus (18 uždavinių × 3);
- Už teisingai išspręstą uždavinį skiriama 6, 9 arba 12 taškų (priklausomai nuo uždavinio sunkumo lygio);
- Už neišspręstą uždavinį – 0 taškų;
- Už klaidingą atsakymą atimamas trečdalis uždaviniui skiriamų taškų, t. y., atitinkamai 2, 3 arba 4 taškai.

Lietuvoje XVIII informatikos ir informatinio mąstymo konkurso „Bebras“ II etape dalyvavo 287 jauniai (9–10 klasių mokiniai) ir 269 kolegos (11–12 klasių mokiniai).

Lentelėje pateikiamas XVIII konkurso II etapo uždavinių skirstymas pagal amžiaus grupes. Skaičius langelyje prie uždavinio nurodo sudėtingumo lygį:

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

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



Nr.	Uždavinio pavadinimas	Uždavinio identifikatorius	Jauniai	Kolegos
1	Ką tai daro?	2021-PL-03a	6	6
2	Slaptasis skaičius	2021-UY-04	6	6
3	Ado rutuliukų mašina	2021-AU-04b	6	
4	Paveikslo vagystė	2021-SI-03	6	
5	Abėcėlės tvarkos šifras	2021-TR-03	6	
6	Kaladėlių bokštai	2020-PT-02c	9	6
7	Kasos parduotuvėje	2021-PK-08	9	6
8	Paukščių migracija	2021-AU-03	9	
9	Rąstai	2021-EE-03	9	
10	Palygink	2021-LT-05	9	
11	Paštininkė Angelė	2020-CY-01	12	9
12	Paskutinis lapas	2021-KR-06	12	9
13	Išsaugok medžius!	2021-RU-01	12	9
14	Maisto produktų pirkimas	2021-TW-05b	12	9
15	Kraujo tyrimas	2020-IE-07	12	12
16	Bebrų rikiuotė	2021-SP-01		6
17	Gamybos vienetai	2020-IT-04		9
18	Matematinė mašina	2020-DE-06b		12
19	Tylusis kalbėjimas	2020-RU-01		12
20	Tiuringo mašinos	2021-AT-07		12
21	Suoliukų dirbtuvės	2021-LV-03		12

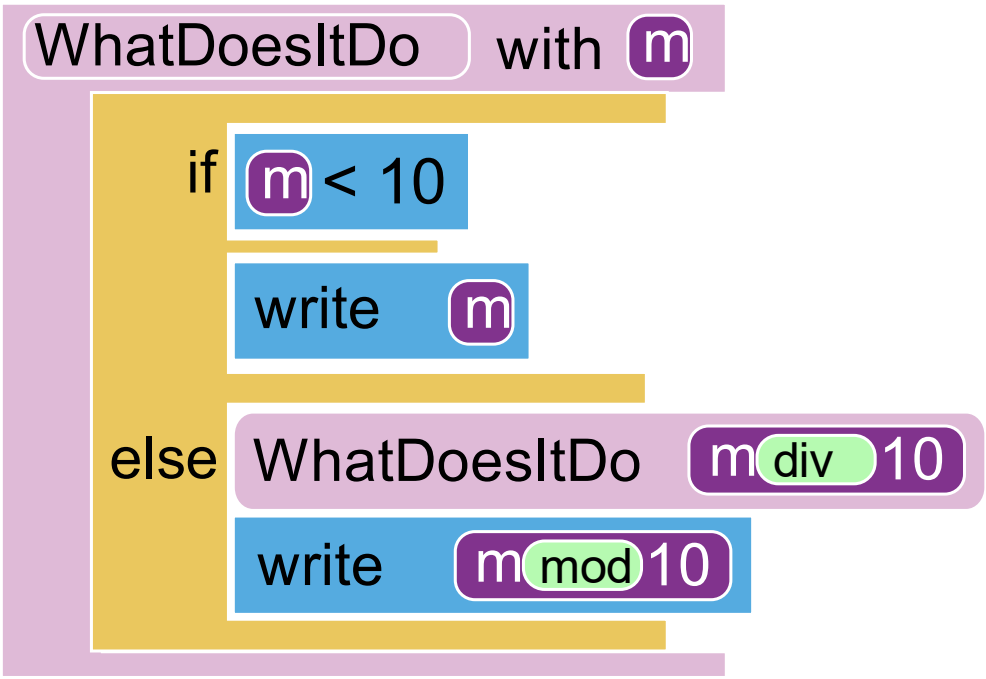
	Jauniai	Kolegos
bebras	6	6

1. Ką tai daro?

Mažasis Bebras sužinojo, kad dalijant sveikuosius skaičius gaunami du rezultatai: sveikoji dalmens dalis ir liekana. Pirmuoju atveju dalybos operacija žymima `div`, o antruoju – `mod`.

Pavyzdžiui, 90 `div` 7 = 12, 90 `mod` 7 = 6.

Mažasis Bebras tyrinėja funkciją „WhatDoesItDo“, kurioje įvedinys *m* yra sveikasis skaičius:



Įrašykite skaičių, kuri išspausdina funkcija „WhatDoesItDo“, kai įvedinys *m* = 30241.

Paaiškinimas

Ši funkcija yra rekursinės funkcijos pavyzdys. Programavime rekursija vadinamas funkcijos kreipimasis į save pačią.

Funkcijos „WhatDoesItDo“ viduje yra kreipimasis į save pačią. Kiekvieną kartą kreipiantis sukuriamas naujas funkcijos egzempliorius. Kai paskutiniame egzemplioriuje gaunama *m* reikšmė, lygi 3, funkcijos rastos reikšmės perduodamos į jų kreipinius, po to visi funkcijos egzemplioriai pašalinami iš kompiuterio atmintinės atvirkščia tvarka, negu ten jie buvo įrašyti. Pirmiausia bus spausdinamas trejetas, nes tai buvo penkto egzemplioriaus rezultatas, toliau bus nulis – ketvirto egzemplioriaus rezultatas ir t. t.

Lentelėje pažingsniui vaizduojama, koks įvedinys pateikiamas funkcijai, į ją kreipiantis 5 kartus, ir kaip iš funkcijos egzempliorių „surenkamas“ rezultatas.

	Input (įvestis)	p(m<10)	m div 10	m mod 10	Output (išvestis)
1	30241	False	3024	–	–
2	3024	False	302	–	–
3	302	False	30	–	–
4	30	False	3	–	–
5	3	True	–	–	3
4	30	–	–	0	0
3	302	–	–	2	2
2	3024	–	–	4	4
1	30241	–	–	1	1

Tai informatika!

Rekursinės funkcijos leidžia veiksmus kartoti nenaudojant ciklo. Jeigu kreipinys į funkciją yra jos pačios viduje, tai reiškia, kad ten bus pakartotinai atliekami jos pačios veiksmai, tik galbūt jau su kitais duomenimis. Taigi rekursija išreiškia veiksmų kartojimą.

Bet kokiai veiksmų atlikimo tvarkai aprašyti pakanka trijų valdymo struktūrų: komandų (sakinių) sekos, pasirinkimo komandos ir ciklo. Ciklą pakeitę rekursija, gauname kitą trejetą – komandų seką, pasirinkimo komandą ir rekursiją. Šiomis konstrukcijomis (valdymo struktūromis) taip pat galima išreikšti veiksmų atlikimo tvarką.

Kadangi rekursija yra abstraktesnė už ciklą konstrukcija, ją labiau mėgsta teoretikai. Ciklus vertina praktikai, nes juose konkrečiau nurodoma veiksmų atlikimo tvarka. Programos su ciklais būna ekonomiškesnės negu tuos pačius veiksmus aprašančios rekursinės programos. Teoriškai kiekvieną rekursiją galima pakeisti ciklu ir atvirkščiai.

Nuorodos:

http://www.ragaine.su.lt/mokomoji/pascal/Paskal8.htm

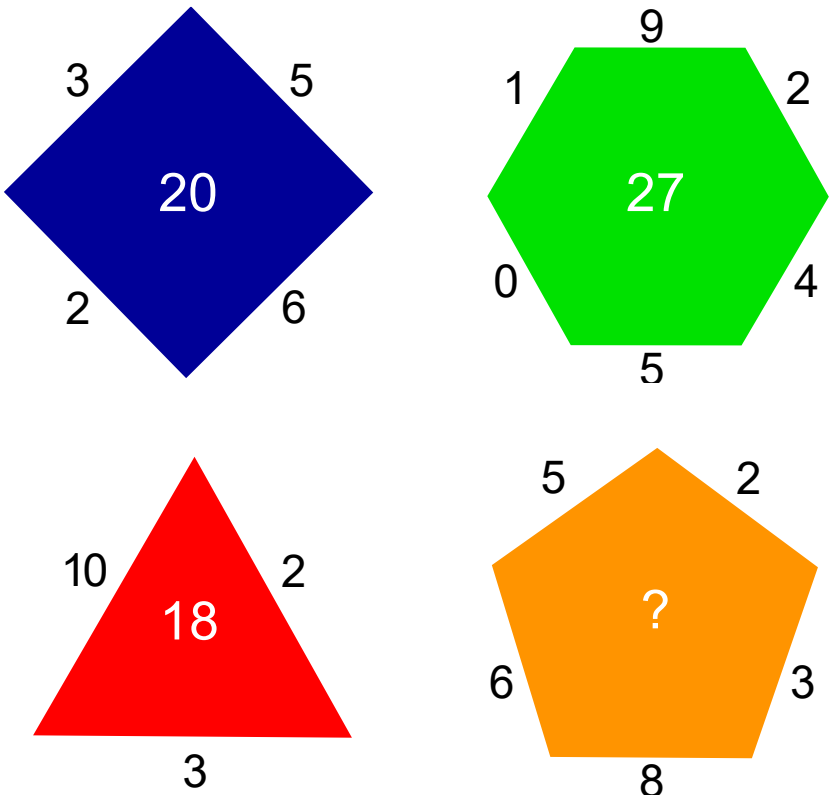
https://bebras.lt/wp-content/uploads/2015/09/Uzdaviniu-2015-m.-knygele.pdf (54 p.)

https://bebras.lt/wp-content/uploads/2017/01/Algorithms-LT-v4.pdf (20 p.)

	Jauniai	Kolegos
	6	6

2. Slaptasis skaičius

Bebrų pasaulyje atsiskaitoma specialiomis monetomis. Kiekvienos formos monetos vertė unikali ir užrašyta monetos centre.



Kokia penkiakampės monetos vertė?

A) 22

B) 23

C) 26

D) 29

Paaiškinimas

Teisingas atsakymas: 29.

Kiekvienos formos monetos vertė apskaičiuojama šitaip: susumuojami visi skaičiai, užrašyti už monetos ribų, ir pridedamas monetos figūros kraštinių skaičius. Šioje užduotyje pateiktos keturių skirtingų formų monetos. Jų vertės:

Mėlynas kvadratas: 3 + 5 + 6 + 2 + 4 = 20

Raudonas trikampis: 10 + 2 + 3 + 3 + 3 = 18

Žalias šešiakampis: 1 + 9 + 2 + 4 + 5 + 0 + 6 = 27

Oranžinis penkiakampis: 5 + 2 + 3 + 8 + 6 + 5 = 29

Tai informatika!

Kiekvienos monetos vertę sudaro dvi dalys. Svarbu išsiaiškinti, kad monetos vertė (baltas skaičius centre) gaunama sudėjus kiekvienos monetos išorėje esančius skaičius ir dar pridėjus monetos figūros kraštinių skaičių. Visų monetų verčių skaičiavimai atliekami pagal tą pačią schemą.

Raštas (šablonas) ir jo atpažinimas naudojamas inžinerijoje, kompiuterijoje, matematikoje ir yra susijęs su fiziniais arba abstrakčiais objektais. Norint atpažinti šablonus, pirmiausia reikia atlikti analizę (pavyzdžiui, atliekant atidų stebėjimą, dėsningumų paiešką). Šablonai gali būti gaunami atliekant segmentavimą, charakteristikų išskyrimą ir aprašymą, kai išskiriamos ir aprašomos kiekvieno objekto savybės. Šios analizės tikslas – išgauti duomenis, kurie leistų atpažinti savybes, atrasti dėsningumus objektų rinkiniuose. Kompiuterių moksle taikomas programinės įrangos projektavimo šablonas yra bendras sprendimas, kuris padeda pagreitinti kompiuterių programų kūrimą.

3. Ado rutuliukų mašina

Inžinierius Adas buvo paprašytas sukurti mašiną rutuliukams rūšiuoti.

Adas žino šiuos rutuliukų dizaino ribojimus:

1. Iš metalo pagaminti rutuliukai negali būti dideli.
2. Iš akmens pagaminti rutuliukai negali būti raudoni.
3. Dideli rutuliukai negali būti papuošti blizgučiais.
4. Raudoni rutuliukai negali būti papuošti mozaika.

Kuris rutuliukų dizainas neprieštarauja numatytiems dizaino ribojimams? (dydis, spalva, medžiaga, puošyba)

Atsakymas	Dydis	Spalva	Medžiaga	Puošyba
A	Mažas	Raudonas	Akmuo	Blizgučiai
B	Mažas	Raudonas	Metalas	Mozaika
C	Didelis	Raudonas	Metalas	Mozaika
D	Didelis	Geltonas	Akmuo	Blizgučiai

Paaiškinimas

Teisingas atsakymas: D.

Taikydami ribojimus vieną po kito, galime pašalinti neleidžiamus derinius, todėl lieka tik šeši galimi variantai.

Vienas iš būdų rasti leidžiamą dizainą – iš eilės nagrinėti kiekvieną dizaino aspektą. Jei pasirenkame mažą dydį, tai 1-as ir 3-ias ribojimai galimų variantų nesumažina. Jei pasirenkame geltoną spalvą, 2-as ir 4-as ribojimai taip pat nedaro įtakos, taigi gauname 4 leistino dizaino variantus (2 medžiagos x 2 papuošimai). Jei vietoj geltonos pasirenkame raudoną spalvą, tai pagal 2-ą ribojimą galėsime naudoti tik metalą, o pagal 4-ą ribojimą – tik blizgučius.

Jei pasirenkame didelį dydį, pagal 1-ą ribojimą galime naudoti tik akmenį, pagal 3-ią ribojimą – tik mozaiką, o pagal 2-ą ir 4-ą ribojimus - tik geltoną spalvą, taigi gauname tik vieną dizaino variantą.

Tai informatika!

Ši užduotis paremta rūšiavimu, t. y., objektų skirstymu į kategorijas pagal tam tikrus kriterijus. Yra keturi kriterijai, pagal kuriuos galime rūšiuoti rutuliukus (dydis, spalva, medžiaga, papuošimai) – taikomi po vieną arba deriniuose.

Taip pat susiduriame su ribojimais, kurie yra mūsų sprendimų ar metodų suvaržymai. Šiuo atveju yra keturi ribojimai, kurie taikomi rutuliukų dizainui. Dėl jų poveikio ne visi dizaino variantai yra galimi.

Skaitmeninis rūšiavimo su ribojimais pavyzdys yra filtrų naudojimas poaibių paieškai tam tikrame duomenų rinkinyje. Taip galima filtruoti skaičiuoklėse, kur filtras taikomas tam tikroms eilutėms paslėpti ar pasirinkti. Be to, filtruoti galima ir paieškos rezultatus, pavyzdžiui, gautus iš internetinės paieškos ar prekybos sistemos, kur filtras gali patikslinti rezultatus, kad jie atitiktų norimus kriterijus.

4. Paveikslo vagystė

Logistikos įmonė „TransArt“ specializuojasi paveikslų pervežimo srityje. Paveikslai atgabenami į parduotuvę apžiūrai. Vėliau kurjeriai gabena juos į galutinę paskirties vietą. Kiekvienas naujas paveikslas dedamas ant jau atvežtų paveikslų viršaus. Kiekvienas kurjeris paima pirmą viršutinį paveikslą ir veža į galutinę paskirties vietą. Visus atvežamus ir išvežamus paveikslus saugumo sumetimais įmonė tiksliai registruoja žurnaluose.

Atvežti į parduotuvę paveikslai	
Laikas	**Paveikslas**
11.40	Bebrai pievoje
12.15	Laimingas bebras
12.55	Saulė ir Mėnulis
13.30	Užburtas miškas
14.18	Ąžuolas ir beržas
15.10	Pelkių romantika

Paveikslai, išvežti iš parduotuvės	
Laikas	**Kurjeris**
12.25	Agrastas
13.35	Cikada
14.35	Agrastas
14.40	Burokas
15.20	Cikada
15.35	Deimantas

Vakare „TransArt“ įmonei buvo pranešta, kad paveikslas „Saulė ir Mėnulis“ taip ir nepasiekė muziejaus, kuris turėjo jį gauti. Kurjeris, paėmęs šį paveikslą iš parduotuvės, tikriausiai jį pavogė!

Kas pavogė paveikslą „Saulė ir Mėnulis“?

Paaiškinimas

Yra dvi svarbios įvykių rūšys. Kažkas padeda paveikslą ant dėklo (angl. *stack*) ir kažkas paima paveikslą nuo dėklo viršaus. Iš užduoties lentelių sukuriame naują lentelę, kurioje rodomi įvykiai ir dėklo būsenos, surikiuotos pagal laiką.

Laikas	Įvykis	Paveikslai dėkle
11.40	Atvežamas paveikslas „Bebrai pievoje“	Bebrai pievoje
12.15	Atvežamas paveikslas „Laimingas bebras“	Laimingas bebras Bebrai pievoje
12.25	Agrastas paima paveikslą „Laimingas bebras“	Bebrai pievoje
12.55	Atvežamas paveikslas „Saulė ir Mėnulis“	Saulė ir Mėnulis Bebrai pievoje
13.30	Atvežamas paveikslas „Užburtas miškas“	Užburtas miškas Saulė ir Mėnulis Bebrai pievoje
13.35	Cikada paima paveikslą „Užburtas miškas“	Saulė ir Mėnulis Bebrai pievoje
14.18	Atvežamas paveikslas „Ąžuolas ir beržas“	Ąžuolas ir beržas Saulė ir Mėnulis Bebrai pievoje
14.35	Agrastas paima paveikslą „Ąžuolas ir beržas“	Saulė ir Mėnulis Bebrai pievoje
14.40	Burokas paima paveikslą „Saulė ir Mėnulis“	Bebrai pievoje

Tai informatika!

Šioje užduotyje yra trys informatikos idėjos. Viena iš jų yra dėklo (angl. *stack*) sąvoka. Dėklas – tai duomenų struktūra, organizuota taip, kad paskutinis į dėklą įdėtas elementas būtų pirmasis iš jo paimtas („Paskutinis į maišą – pirmas iš maišo“). Antroji – sujungimas (suliejimas, sąlaja, angl. *merging*): sprendimas gaunamas, paėmus du surikiuotus (pagal laiko žymas) įvykių sąrašus, sujungiant juos į vieną surikiuotą sąrašą, kuris pateiktas sprendime. Tai greičiausias duomenų rikiavimo būdas – sąlajos rikiavimo algoritmas.

Trečioji idėja – visa įvykių seka yra tarsi programos vykdymo modelis. Paveikslo vagystė yra įvykis, dėl kurio sustabdoma („sugenda“) kompiuterio vykdoma programa. Tai vadinama išimtinė situacija. Norint surasti išimties priežastį (netinkamai besielgiantį kurjerį ar netinkamai veikiančias programos eilutes), reikia sekti programos vykdymą tol, kol surandama programos trikties vieta. Tai vadinama sekimu (angl. *tracing*). Radęs tą vietą programuotojas ieško sprendimo, kaip elgtis su išimtimi, kad programa nenutrūktų („nenulūžtų“).

Nuorodos:

https://lt.wikipedia.org/wiki/S%C4%85lajos_rikiavimo_algoritmas

https://lt.wikipedia.org/wiki/Rietuv%C4%97

https://en.wikipedia.org/wiki/Stack_(abstract_data_type)

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

5. Abėcėlės tvarkos šifras

Marsiečiai sukūrė šifravimui skirtą algoritmą. Užšifruotas žodis susideda iš dviejų dalių: pirmoji – tai šifruojamo žodžio skaitinė reikšmė, antroji – kiekvienos žodžio raidės vieta, tas raides išrikiavus pagal abėcėlę.

Visoms šifravimo operacijoms jie naudoja tokią lentelę:

A	B	M	N	O	R	S	T	U
1	2	4	10	50	180	300	650	960

Planetos pavadinimas MARS užšifruojamas taip: skaitinė žodžio reikšmė gaunama susumavus raidžių vertes, nurodytas lentelėje (4+1+180+300 = 485), o antroji dalis – nustatant raidžių eilę pagal abėcėlę (A-M-R-S), tad raidžių eilės indeksai bus: A=1, M=2, R=3, S=4.

Visas žodis MARS, jį užšifravus, atrodys taip: 485;2134.

Jeigu žodis SATURN būtų šifruojamas naudojantis tuo pačiu algoritmu, kuris iš duotų atsakymų būtų teisingas?

A) 1440;415632

B) 1440;718964

C) 2101;415632

D) 2101;718964

Paaiškinimas

Teisingas atsakymas: C.

Jį galite rasti net ir neskaičiuodami. Pirmiausia patikriname pirmąją užšifruoto žodžio dalį – šifruotų raidžių sumą. Galimi du atsakymai: 1440 ir 2101. Žodyje SATURN yra raidės T ir U – iš dešinės lentelės pusės su labai dideliais skaičiais 650 ir 960. Matyti, kad bandant paimti vien šių dviejų skaičių sumą gauname vertę, didesnę nei 1440. Iš to darome išvadą, kad teisinga suma turi būti 2101.

Antrajai užšifruoto žodžio daliai pateikiamos tik dvi galimybės: 415632 ir 718964. SATURN turi tik 6 raides, o raidžių indeksai šifre reiškia raidžių vietas. Todėl negali būti 7 ar didesnio indekso (žodyje SATURN nėra raidės 7-oje vietoje). Taigi 718964 yra neteisingas, o 415632 turi būti teisingas.

A atsakymas neteisingas, nes pirmoji dalis yra neteisinga.

D atsakymas neteisingas, nes antroji dalis yra neteisinga.

B atsakymas neteisingas, nes abi dalys neteisingos.

Galime suskaičiuoti ir įrodyti, kad C atsakymas tikrai teisingas:

Pirma dalis:

S = 300, A = 1, T = 650, U = 960, R = 180, N = 10.

Bendra vertė: 300+1+650+960+180+10 = 2101.

Antra dalis:

abėcėlinė raidžių tvarka žodyje tokia: S=4, A=1, T=5, U=6, R=3, N=2,

todėl gauname 415632.

Taigi žodis SATURN, jį užšifravus, bus pavaizduotas taip: 2101; 415632.

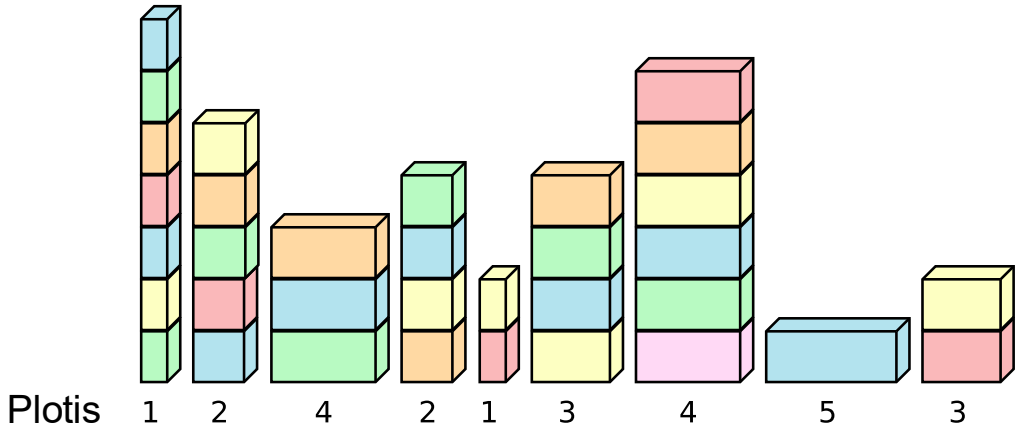
Tai informatika!

Šioje užduotyje naudojama perkėlimo šifro idėja kartu su kontroline suma. Naudojant perkėlimo šifrą raidės sukeičiamos vietomis, o indeksas nurodo raidžių eiliškumą. Kontrolinės sumos padeda nustatyti klaidas, atsiradusias dėl trukdžių perduodant šifruotus pranešimus.

	Jauniai	Kolegos
	9	6

6. Kaladėlių bokštai

Bebras Simas žaidžia su savo kaladėlėmis. Visos kaladėlės yra vienodo aukščio, bet skirtingo pločio. Simas pastatė devynis gražius bokštus, kurių kiekvienas sudarytas iš tokio paties pločio kaladėlių. Yra dar likusių kaladėlių.



Yra du būdai pakeisti bokšto aukštį: pridėti kaladėlių ant viršaus arba nuimti kaladėles nuo viršaus. Bet kuriuo atveju energijos sąnaudos keičiant bokšto aukštį yra proporcingos kaladėlių pločiui ir jų skaičiui. Pavyzdžiui, 2 kaladėlėms nuimti nuo bokšto, kurio plotis 1, sunaudojama 2×1 = 2 energijos vienetai, o 4 kaladėlėms uždėti ant bokšto, kurio plotis 3, sunaudojama 4×3 = 12 energijos vienetų.

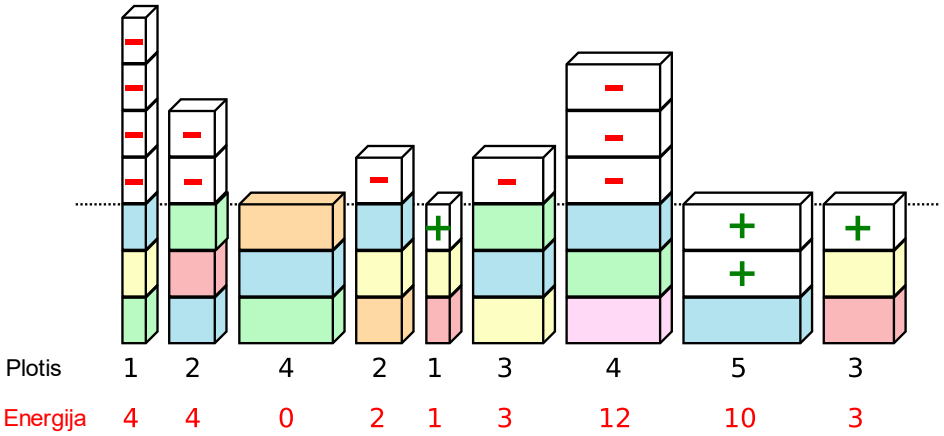
Simas nori, kad visi bokštai būtų vienodo aukščio, taip pat jis nori sunaudoti kuo mažiau energijos.

Kiek mažiausiai energijos reikia Simui, kad galėtų suvienodinti visų bokštų aukštį?

Paaiškinimas

Teisingas atsakymas: 39.

Jei kaladėlių pridėjimą vaizduosime  , o nuėmimą  tada paveiksle rodomas teisingas sprendimas ir visi bokštai yra 3 kaladėlių aukščio:



Vidutinis visų devynių bokštų aukštis yra 4 (1 2 2 3 4 4 5 6 7). Galima manyti, kad vidutinis aukštis yra atsakymas, tačiau šiuo atveju taip nėra. Kaip apskaičiuota pateikiamoje lentelėje, visi kiti galimi aukščiai pareikalautų didesnių išlaidų:

	Dabartinis aukštis	7	5	3	4	2	4	6	1	2	*Sunaudojama energija*
	Plotis	1	2	4	2	1	3	4	5	3	
Sunaudojama energija	Aukštis 1	6	8	8	6	1	9	20	0	3	61
	Aukštis 2	5	6	4	4	0	6	16	5	0	46
	Aukštis 3	4	4	0	2	1	3	12	10	3	39
	Aukštis 4	3	2	4	0	2	0	8	15	6	40
	Aukštis 5	2	0	8	2	3	3	4	20	9	51
	Aukštis 6	1	2	12	4	4	6	0	25	12	66
	Aukštis 7	0	4	16	6	5	9	4	30	15	89

Taip yra dėl to, kad blokai nėra vienodo pločio. Tačiau energijos sąnaudas nustatanti funkcija yra vienarūšė (turi vieną minimumą), todėl užuot bandę visus įmanomus aukščius, galime naudoti protingesnę strategiją minimumo paieškai, pavyzdžiui, trinarę paiešką. Taikydami trinarę paiešką galime pašalinti du didžiausius aukščius ir du žemiausius aukščius iš septynių aukščių, palikdami aukščius 3, 4 ir 5. Tada tereikia apskaičiuoti energijos sąnaudas, kurios sudaro šiuos tris aukščius.

Tai informatika!

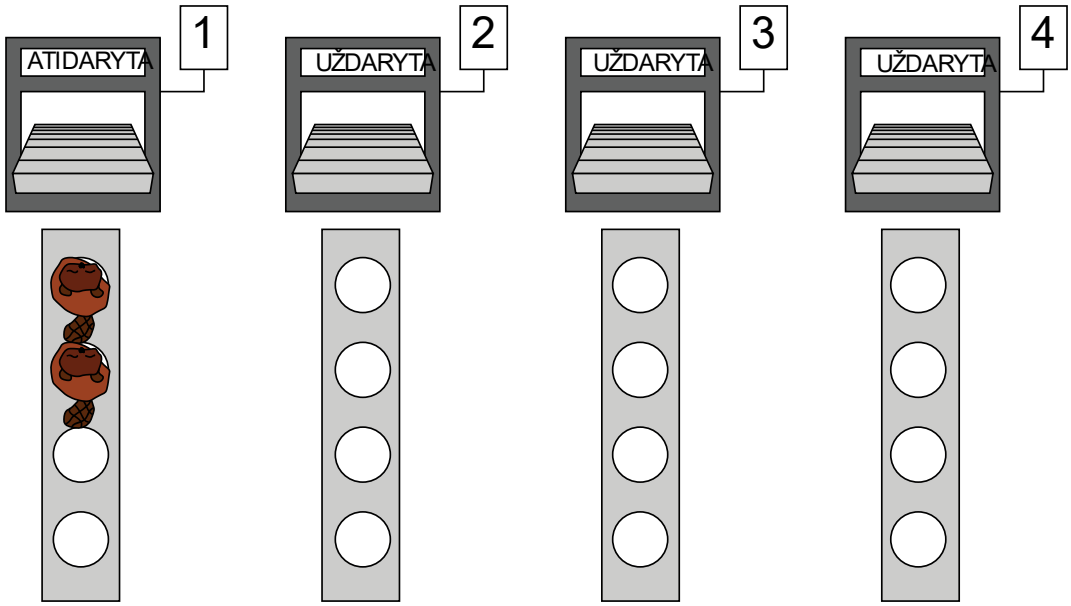
Kiekvieną kaladėlių krūvą kompiuterių moksle vadiname dėklu (angl. *stack*) su dviem galimomis operacijomis: pridėti (naujos kaladėlės uždėjimas ant krūvos viršaus) ir paimti (viršutinės kaladėlės pašalinimas). Spręsdami šią užduotį turime atsižvelgti į dabartinį kiekvieno jau pastatyto bokšto aukštį ir plotį.

Dėklai yra labai naudingi daugeliui algoritmų, sudėtingesnių nei tiesiog bokštų statymas. Pavyzdžiui, jie gali būti naudojami aritmetiniams reiškiniams įvertinti.

Kompiuterių moksle ieškoti geriausio sprendimo (pavyzdžiui, minimalių sąnaudų ar kainos) yra įprastas dalykas. Tai vadinama optimizavimo uždaviniu. Šiuo atveju, kadangi kaštai turi unimodalinį pasiskirstymą, galėtume naudoti trinarę paiešką, kuri nustato, kad minimumas arba maksimumas negali būti nei pirmame, nei paskutiniame domeno trečdalyje, o tada kartojasi likusiuose dviejuose trečdaliuose. Trinarė paieška yra „skaldyk ir valdyk“ algoritmo pavyzdys.

7. Kasos parduotuvėje

Parduotuvėje yra keturios sunumeruotos kasos: 1, 2, 3, 4. Prie kiekvienos kasos gali susidaryti ne daugiau kaip 4 pirkėjų eilė, įskaitant aptarnaujamą pirkėją. Kiekvienoje kasoje vienu metu aptarnaujamas vienas pirkėjas. Aptarnauti vieną pirkėją užtrunka 2 minutes. Iš pradžių atidaryta tik pirma (1) kasa.



Norėdamas susimokėti už pirkinius, pirkėjas atsistoja į eilę prie pirmos kasos, jei eilėje yra vietos. Pradžioje bando 1-ą kasą, tada 2-ą kasą ir t. t.

Jei atidarytų kasų eilėse nebėra vietos, atidaroma nauja kasa, ir pirkėjas nueina prie jos. Kasai pradėti darbą užtrunka 1 minutę, todėl ką tik atidarytoje kasoje aptarnauti pirmą pirkėją prireiks 3 minučių. Kiekvienas tolesnis pirkėjas aptarnaujamas per 2 minutes, kaip įprasta.

Jei tam tikru momentu yra jau aptarnautų, bet dar nepalikusių kasos pirkėjų, ir naujų pirkėjų, norinčių atsistoti į eilę, laikysime, kad aptarnauti klientai išeina pirmi, atlaisvindami vietą eilėse naujiems pirkėjams.

Prie kasų ateina 12 pirkėjų: po du kas minutę (t. y., iš pradžių ateina du pirkėjai, po minutės – dar du ir t. t.). **Kiek užtruks aptarnauti juos visus?**

A) 12 minučių

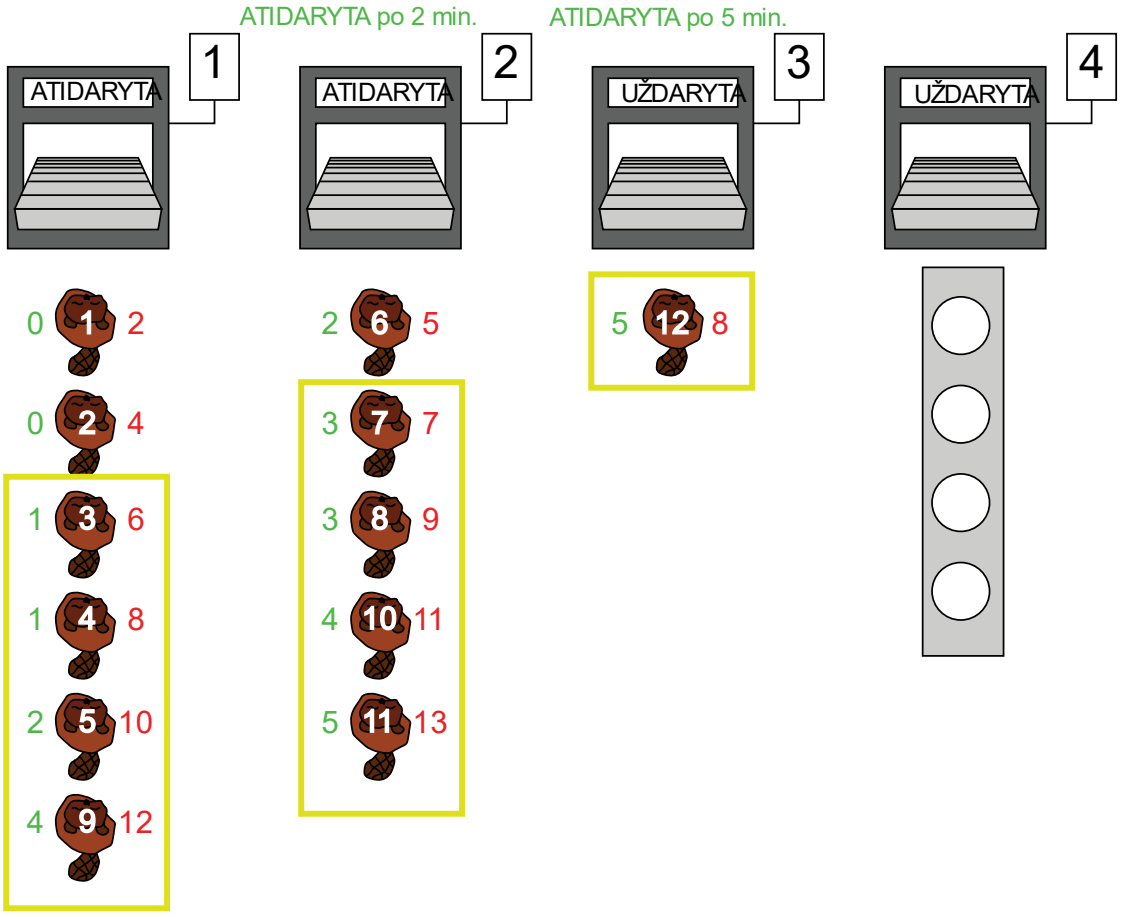
B) 11 minučių

C) 13 minučių

D) 8 minutės

Paaiškinimas

Paveiksle vaizduojamas laikas, kada kiekvienas pirkėjas ateina (žalia spalva) ir kada išeina (raudona spalva):



Kaip matome, išėjimo iš kiekvienos kasos laikai yra numatomi iš anksto, kadangi kiekvienas pirkėjas išeina po 2 minučių nuo ankstesnio aptarnauto. Norėdami išspręsti šį uždavinį, turime sekti šiuos laikus (vaizduojamus raudonu šriftu), kurie gali būti apskaičiuoti kiekvienam ateinančiam pirkėjui.

Po 2 minučių, kai ateina du nauji pirkėjai, galime nebenagrinėti pirmojo pirkėjo prie pirmos kasos (jis jau aptarnautas), vienas naujas pirkėjas, atsistojęs į eilę, bus aptarnautas 2 min. vėliau už ankstesnį, stovintį eilėje (8 + 2 = 10), o kiti nauji klientai ateina į naujai atidarytą antrą kasą ir bus aptarnauti 3 minutėmis vėliau esamo laiko (2 + 3 = 5).

Už 5 minučių eilės bus tokios, kokios pavaizduotos oranžiniuose stačiakampiuose. Tačiau turėsime palaukti, kol bus aptarnautas paskutinis pirkėjas, – tai bus paskutinis iš antros eilės, 13 minutę.

Taigi teisingas atsakymas yra 13 minučių.

A) atsakymas – tai laikas, kai paskutinis klientas išeina iš pirmos eilės.

D) atsakymas – tai laikas, kai paskutinis pirkėjas palieka trečią eilę.

Tai informatika!

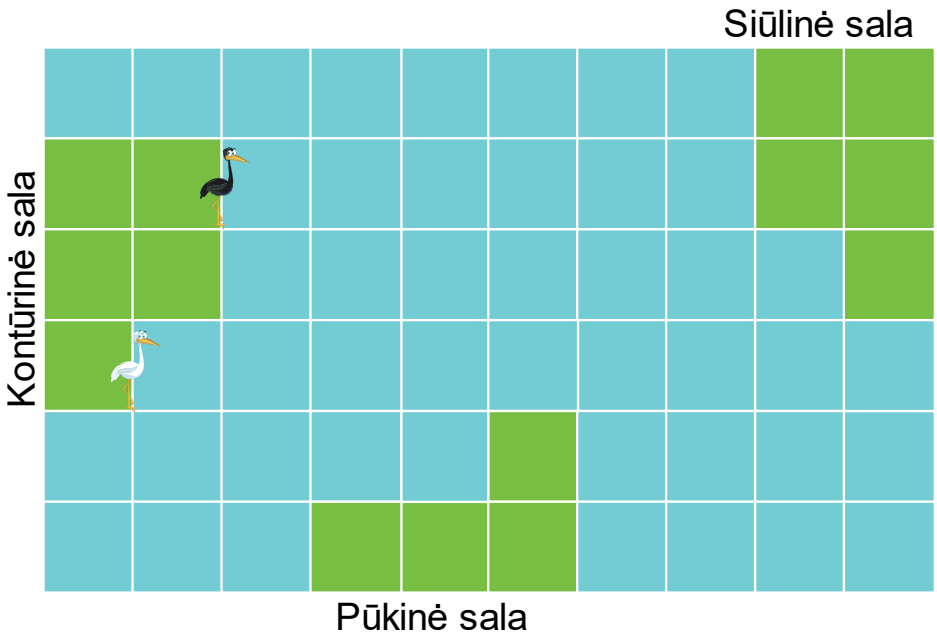
Debesijos paslaugos, pvz., „Google Cloud“, „Amazon Web Services“, „Microsoft Azure“ ir kt. dinamiškai pritaiko kompiuterinius išteklius, remiantis klientų poreikiais. Taip klientai moka tik už išteklius, kurių jiems reikia. Toks lankstus išteklių suteikimas vadinamas dinaminiu paskirstymu.

Ši užduotis iliustruoja paprastą dinaminio paskirstymo atvejį. Čia ištekliai – tai kasos. Kasos atidarymas reiškia darbo laiko apmokėjimą dar vienam darbuotojui. Todėl kasos atidaromos pagal poreikį, didėjant pirkėjų skaičiui. Tai sumažina aptarnavimo kainą, kai paklausa maža.

Praktikoje dinaminis paskirstymas apima ir išteklių mažinimą mažėjant paklausai. Šio uždavinio kontekste tai būtų kasų uždarymo strategija, kai pirkėjų skaičius pradeda mažėti.

8. Paukščių migracija

Trijų salų pavadinimai – Kontūrinė, Pūkinė ir Siūlinė – susiję su paukščių plunksnų tipais. Kontūrinėje saloje šiuo metu yra du paukščiai, norintys migruoti į šiltesnę Siūlinę salą žiemoti.



Juodasis garnys gali skristi virš jūros 2 kvadratų per valandą greičiu ir, perskridęs 4 kvadratus, turi 1 valandą ilsėtis sausumoje.

Baltasis ibis gali skristi virš jūros 4 kvadratų per valandą greičiu, tačiau, perskridęs 4 kvadratus, turi 2 valandas ilsėtis sausumoje.

Abu paukščiai taip pat gali eiti sausuma 1 kvadrato per valandą greičiu. Kiekvienas paukštis gali judėti tik kairėn ar dešinėn, viršun ar apačion pagal pateiktą žemėlapį – įstrižai judėti negalima.

Kuris paukštis iš Kontūrinės salos į Siūlinę salą nuskrenda greičiausiai ir koks yra jų greičiausio skrydžio trukmės skirtumas?

A) Juodasis garnys nuskris 1 valanda greičiau.

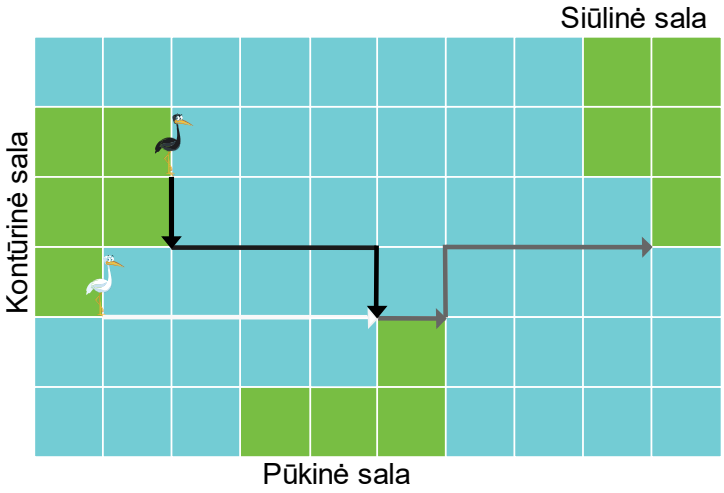
B) Baltasis ibis nuskris 1 valanda greičiau.

C) Juodasis garnys nuskris 2 valandomis greičiau.

D) Baltasis ibis nuskris 2 valandomis greičiau.

Paaiškinimas

Atsakymas: D. Baltasis ibis nuskris 2 valandomis greičiau.



Pirmiausia turime rasti kiekvieno paukščio greičiausią kelią, po to palyginti gautus laikus.

Baltasis ibis iš Kontūrinės salos nuskrenda 4 kvadratus dešinėn į Pūkinę salą, tai užtrunka 1 valandą. Pailsėjęs 2 valandas, jis paeina 1 kvadratą sausuma į dešinę ir tai užtrunka 1 valandą. Galiausiai jis skrenda 4 kvadratus iš Pūkinės salos į Siūlinę salą, tai trunka 1 valandą. Bendras baltojo ibio skrydžio laikas yra 1 + 2 + 1 + 1 + 1 = 5 valandos.

Juodasis garnys pirmiausia eina 1 kvadratu žemyn sausuma, tai užtrunka 1 valandą. Iš čia jis skrenda 4 kvadratus iki Pūkinės salos ir tai užtrunka 2 valandas. Pailsėjęs 1 valandą, juodasis garnys eina 1 kvadratą į dešinę, tai trunka 1 valandą. Galiausiai jis skrenda 4 kvadratus link Siūlinės salos, tai trunka 2 valandas. Bendras juodojo garnio skrydžio laikas yra 1 + 2 + 1 + 1 + 1 + 1 + 2 = 7 valandos. Taigi baltasis ibis iš tiesų yra greitesnis: jis nuskrenda 2 valandomis greičiau, nei juodasis garnys.

Atkreipiame dėmesį, kad abiejų paukščių skrydžio atkarpos virš jūros yra kiek įmanoma ilgiausios, kiek jie gali skristi be poilsio. Tai reiškia, kad nė vieno (lėtesnio) skrydžio atkarpos negalime pakeisti (greitesniu) skrydžiu, nes tada skrydžio atstumo neužteks sausumai pasiekti. Tai reiškia, kad mūsų apskaičiuotas bendrasis kelionės laikas yra optimalus.

Tai informatika!

Šiame uždavinyje nagrinėjama optimizavimo problema, t. y., procesas, kai ieškomas ir randamas veiksmingiausias sprendimas. Šiuo atveju kiekvienam paukščiui reikia rasti tiesiausią kelią iš vienos salos į kitą. Tačiau turime atsižvelgti į pateiktus ribojimus, t. y., jie turi įtakos, varžo mūsų sprendimus ar metodus. Mūsų uždavinyje kiekvienas paukštis, įveikęs tam tikrą atstumą, turi pailsėti, o tai riboja, kiek ilgai paukštis gali judėti nepertraukiamai.

Kitas optimizavimo pavyzdys galėtų būti užduotis pagaminti kuo didesnę dėžę naudojant tam tikrą kiekį medžiagos, tarkime, skardos. Norėtume padidinti dėžės tūrį (optimizavimas), tačiau mus riboja turimos medžiagos, su kuria turime dirbti, kiekis (ribojimas).

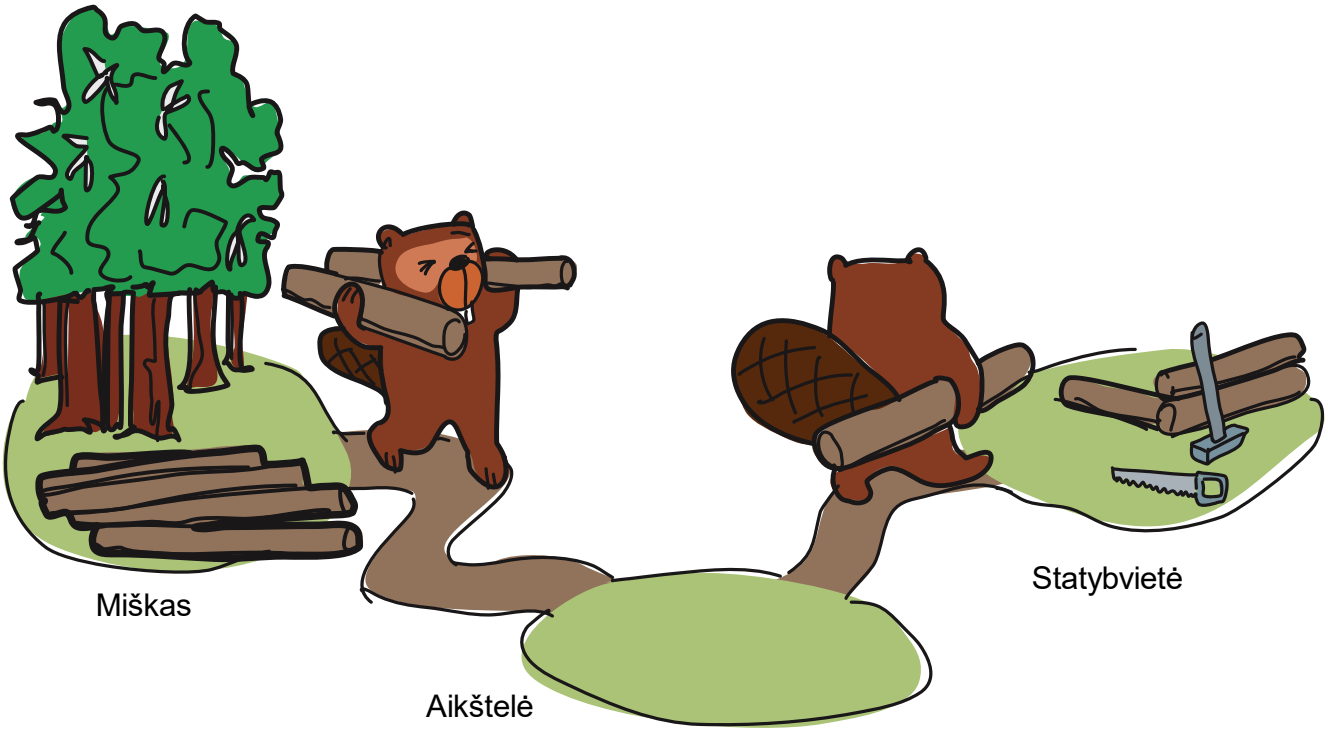
Technologinis pavyzdys: sprendžiama, kaip paskirstyti atmintį įvairioms kompiuterio užduotims vienu metu atlikti. Kompiuteris turi nustatytą atminties kiekį (ribojimas), tačiau galima siekti paskirstyti atmintį įvairioms užduotims taip, kad pagerėtų vartotojo našumas (optimizavimas).

9. Rąstai

Domas ir Simas stato namą iš rąstų. Domas neša rąstus iš miško į saugojimo aikštelę. Vienu metu jis gali nešti du rąstus, o nuo miško iki aikštelės nueiti jam užtrunka 5 minutes. Simas neša rąstus iš aikštelės į statybvietę. Šį atstumą jis gali įveikti per 2 minutes, tačiau gali nešti tik vieną rąstą. Abu bebrai eina tokiu pačiu greičiu nepaisant to, ar neša rąstus, ar ne. Jie dirba tokia tvarka:

- Kai Domas atneša naujų rąstų į aikštelę, juos ten palieka ir, prieš grįždamas į mišką, pašaukia Simą. Simas, išgirdęs Domą, nustoja dirbti statybvietėje ir eina parsinešti rąstų iš aikštelės.
- Kai Simas paima paskutinį rąstą iš aikštelės, grįžęs į statybvietę ten lieka tęsti statybos darbų. Tačiau, jei tai buvo ne paskutinis rąstas aikštelėje, Simas, nunešęs jį į statybvietę, iškart eina paimti kito rąsto.

Kiek daugiausia rąstų bebrai galėtų nunešti į statybvietę aprašyta tvarka dirbdami 30 minučių?



Paaiškinimas

Teisingas atsakymas: 5 .

Per 30 minučių Domas gali triskart nueiti iki aikštelės ir atgal. Į aikštelę po du rąstus jis atneš po 5, 15 ir 25 minučių nuo darbo pradžios.

Kai Domas pirmą kartą atneš rąstus į aikštelę, aikštelė dar tuščia, todėl jis pašauks Simą ir, palikęs atneštus rąstus, grįš į mišką. Išgirdęs Domą Simas užtruks 2 minutes nueiti iki aikštelės, taigi jis pirmąjį rąstą į statybvietę nuneš po 9 minučių nuo darbo pradžios, o antrąjį – po 13 minučių nuo darbo pradžios. Nunešęs antrąjį rąstą jis liks statybvietėje tęsti darbų.

Ta pati situacija pasikartos, kai Domas atneš kitus du rąstus po 15 minučių nuo darbo pradžios. Šiuos rąstus Simas nuneš į statybvietę praėjus 19 ir 23 minutėms nuo darbų pradžios.

Tačiau iš trečiosios poros rąstų, kuriuos Domas atneš į aikštelę, tik pirmasis rąstas spės pasiekti statybvietę iki 30 minučių pabaigos. Taigi iš viso per 30 minučių bebrai į statybvietę gali pristatyti 5 rąstus.

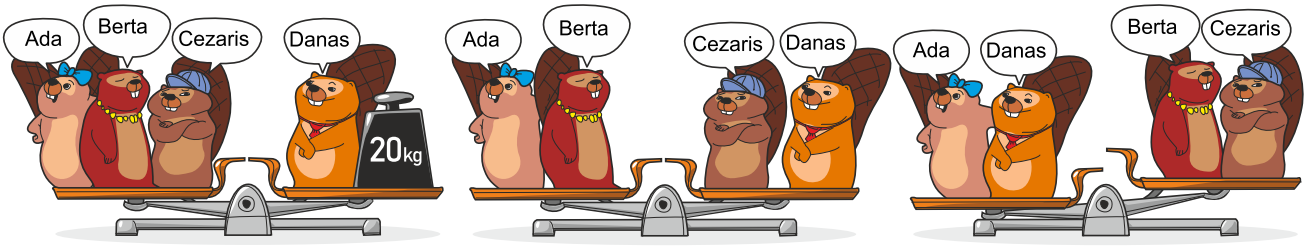
Tai informatika!

Tai, kaip šiame uždavinyje dirba Domas ir Simas, panašu į vadinamąjį gamintojo ir vartotojo modelį, naudojamą kompiuterių sistemose vykdant lygiagrečius skaičiavimus. Šiuo atveju Domas modelyje atitiktų gamintojo, o Simas – vartotojo vaidmenį.

Aikštelė – tai buferis (laikina kompiuterio atminties sritis, reikalinga duomenų mainų procesams suderinti). Pasinaudojus aikštele (buferiu), Domui nereikia laukti, kol Simas ateis paimti iš jo rąstų ir jis gali iškart grįžti į mišką. Šūksnis Simui, kai Domas atneša naujus rąstus į tuščią aikštelę, atitinka signalus, kuriais kompiuterių sistemos leidžia vienoms programoms perduoti informaciją kitoms programoms. Tai leidžia Simui dirbti statybvietėje vietoje to, kad jis visą laiką lauktų aikštelėje. Tačiau, kai Domas pašaukia Simą, praeina šiek tiek laiko, kol Simas ateina iš statybvietės į aikštelę ir tai sukelia rąstų „vėlavimą“ į statybvietę. Analogiški procesai vyksta ir kompiuterių sistemose.

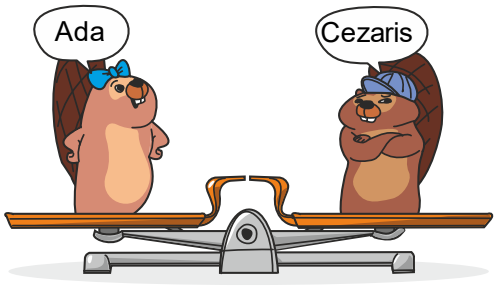
10. Palygink

Keturi bebrai Ada, Berta, Cezaris ir Danas žaidžia su svarstyklėmis ir padarė daug nuotraukų. Pateikiame tris nuotraukas:

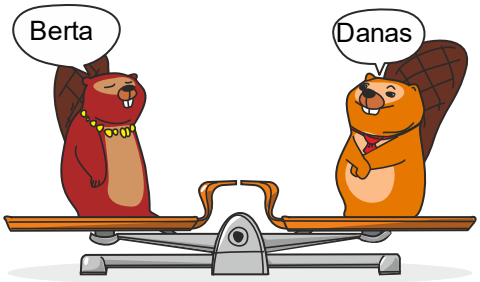


Kuri iš toliau pateiktų nuotraukų taip pat galėjo būti padaryta?

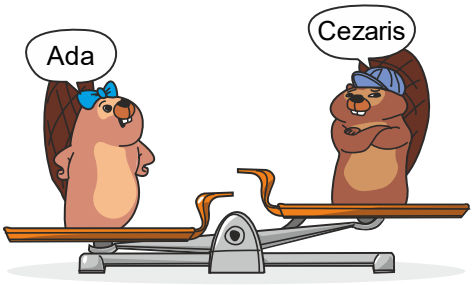
A) Nuotrauka, kurioje Ada sveria tiek pat, kiek ir Cezaris.



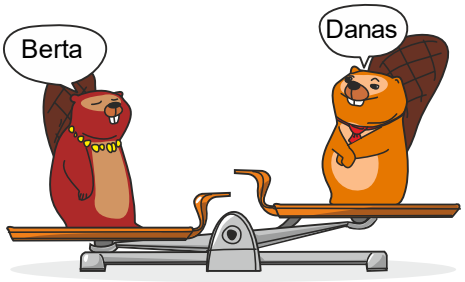
B) Nuotrauka, kurioje Berta sveria tiek pat, kiek ir Danas.



C) Nuotrauka, kurioje Cezaris sveria mažiau nei Ada.



D) Nuotrauka, kurioje Berta sveria daugiau nei Danas.



Paaiškinimas

Galima tik C atsakyme parodyta nuotrauka. Kitų trijų nuotraukų neįmanoma padaryti.

Yra du skirtingi šio uždavinio sprendimo būdai. Šįkart remsimės dedukciniu samprotavimu.

Jei A atsakyme esanti nuotrauka būtų teisinga, tai reikštų, kad Ada sveria tiek pat, kiek ir Cezaris. Tada Berta turi sverti tiek pat, kiek ir Cezaris (iš nuotraukos viršuje dešinėje). Tokiu atveju apatinėje nuotraukoje esančios svarstyklės turėjo būti subalansuotos (Ada+Danas = Berta+Cezaris). Todėl atsakymas A negali būti teisingas.

Jei B atsakyme esanti nuotrauka būtų teisinga, tai reikštų, kad Berta sveria tiek pat, kiek ir Danas. Pirmiau pateiktas argumentas vis tiek galioja. Pagal nuotrauką viršuje dešinėje Ada taip pat turi sverti tiek pat, kiek ir Cezaris. Taigi apatinėje nuotraukoje svarstyklės turėjo būti subalansuotos (Ada+Danas = Berta+Cezaris). Todėl B atsakymo prielaida negali būti teisinga.

Jei D atsakymo nuotrauka būtų teisinga, tai reikštų, kad Berta sveria daugiau nei Danas (Berta > Danas). Vėlgi, pagal nuotrauką viršuje dešinėje, tuomet Cezaris turi sverti daugiau nei Ada (Cezaris > Ada), kad svarstyklės išliktų subalansuotos. Tai reiškia, kad apatinėje nuotraukoje negali būti taip, kad Berta + Cezaris kartu svertų mažiau nei Ada + Danas. Tiesą sakant, turėtų būti atvirkščiai.

Jei C atsakymo nuotrauka teisinga, Cezaris turi sverti mažiau už Adą (Ada < Cezaris). Išlieka įmanoma, kad visos trys pateiktos nuotraukos būtų teisingos. Todėl teisingas atsakymas yra C.

Tai informatika!

Pirma, pagrindinė informatiko kompetencija yra gebėjimas nuspręsti, ar pateiktas pasiūlymas atitinka žinomus apribojimus. Antra, informatikai mokomi iš duotų duomenų (žinomų faktų) išgauti kuo daugiau informacijos. Galiausiai, informatikai yra specialistai, mokantys ieškoti įmanomų duotų uždavinio atvejų sprendimų ir kurti tam tikslui skirtus algoritmus.

Be to, lyginimas labai dažnai naudojamas informatikoje sprendžiant įvairias problemas, pvz., rūšiuojant sąrašus. Palyginimo operatoriai naudojami sąlyginiuose reiškiniuose, kai reikia nuspręsti, kuris kodo blokas bus vykdomas.

Loginis samprotavimas yra labai svarbus informatikams. Programavimo kalbos „Lisp“ ir „Prolog“ ypač gerai tinka simbolinėms manipuliacijoms ir loginiams išvedimams. Iš esmės kompiuterio programa – tai loginių veiksmų, kuriais sprendžiamas uždavinys, įgyvendinimas. Tačiau loginiai samprotavimai svarbūs ne tik programuojant, informatikai dažnai naudoja loginę ir matematinę analizę programos sudėtingumui nustatyti ir programos teisingumui įrodyti.

	Jauniai	Kolegos
bebras	12	9

11. Paštininkė Angelė

Ežere įsikūrusiame kaime yra 6 pastatai, kuriuose bebrai gyvena po vieną arba keliese.

Angelė yra naujoji šio kaimo paštininkė, kuri nieko nežino apie čia gyvenančius bebrus ir nieko apie juos nėra užsirašiusi savo naujoje, tuščioje užrašų knygelėje.

Bebrai bendrauja atvirlaiškiais. Angelė sugalvojo tokią atvirlaiškių pristatymo strategiją:

1. Kiekvieną kartą, kai paštininkei įteikiamas naujas atvirlaiškis, ji užsirašo siuntėjo vardą ir adresą.
2. Tuomet, jei atvirlaiškio gavėjo vardas yra jos užrašų knygelėje, Angelė pristato atvirlaiškį atitinkamu adresu.
3. Jei gavėjo vardo nėra jos užrašų knygelėje, ji pagamina šio atvirlaiškio kopijas ir pristato jas į visus kaimo namus, išskyrus siuntėjo.
4. Bet kuriuo atveju tikrasis atvirlaiškio gavėjas tą pačią dieną atsako atvirlaiškiu siuntėjui (jeigu jo gautas atvirlaiškis nėra atsakymas). Tuomet Angelė savo užrašų knygelėje užsirašo šio bebro vardą bei adresą ir pristato atsakymą nurodytam gavėjui.

Pirmąją Angelės darbo dieną Emilija siunčia atvirlaiškį Jurgiui, o tada Mykolas siunčia atvirlaiškį Emilijai. Antrąją Angelės darbo dieną Sokratas siunčia atvirlaiškį Anastasijai.

Kiek atvirlaiškių Angelė pristatė per dvi pirmąsias savo darbo bebrų kaime dienas?

Paaiškinimas

Teisingas atsakymas: 14 (pristatytų atvirlaiškių).

Pirmiausia, Emilija siunčia atvirlaiškį Jurgiui. Angelė nežino, kur gyvena Jurgis, taigi pristato atvirlaiškio kopiją visais 5-ais adresais (+5 atvirlaiškiai). Jurgis atsako Emilijai ir Angelė pristato atsakymą (+1 atvirlaiškis). Tuomet Mykolas siunčia atvirlaiškį Emilijai. Angelė žino, kur gyvena Emilija, taigi pristato atvirlaiškį jai tiesiogiai (+1 atvirlaiškis). Emilija atsako tą pačią dieną ir Angelė pristato atsakymą tiesiogiai Mykolui (+1 atvirlaiškis). Po pirmosios dienos Angelė yra pristačiusi 8 atvirlaiškius.

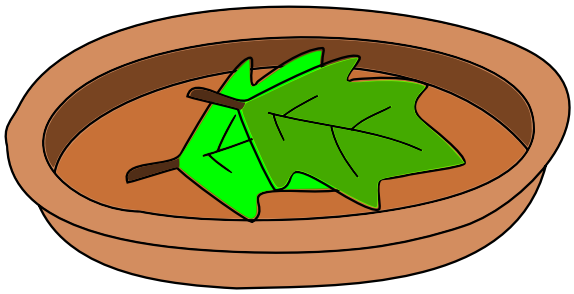
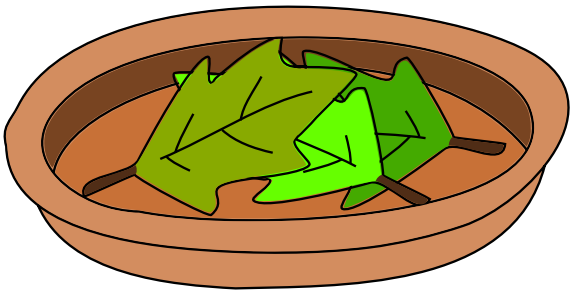
Antrąją dieną Sokratas siunčia atvirlaiškį Anastasijai. Angelė nežino, kur gyvena Anastasija, taigi pristato atvirlaiškio kopiją visais 5-iais adresais (+5 atvirlaiškiai). Anastasija atsako Sokratui ir Angelė pristato atsakymą tiesiogiai Sokratui (+1 atvirlaiškis). Iš viso per šias dvi darbo dienas Angelė pristatė 14 atvirlaiškių.

Tai informatika!

Aprašyta procedūra atitinka metodą, kuriuo remiantis tinklo komutatorius užpildo savo tinklo plokščių (MAC) adresų lentelę.

12. Paskutinis lapas

Emilė ir Justinas žaidžia žaidimą „Paskutinis lapas“. Žaidimas žaidžiamas naudojant dvi krūveles medžių lapų, kaip pavaizduota paveikslėlyje. Žaidėjai pakaitomis ima lapus iš krūvelių, kol lapų nebelieka. Savo eilės metu žaidėjas privalo paimti vieną arba daugiau lapų iš vienos krūvelės. Laimi žaidėjas, paėmęs paskutinį lapą.

	
Ⓐ - 2 lapai	Ⓑ - 3 lapai

Kuriuo iš nurodytų atvejų Emilė galėtų laimėti, nepriklausomai nuo Justino veiksmų?

A) Emilė pradeda paimdama 2 lapus iš Ⓐ krūvelės.

B) Emilė pradeda paimdama 2 lapus iš Ⓑ krūvelės.

C) Justinas pradeda paimdamas 1 lapą iš Ⓐ krūvelės.

D) Justinas pradeda paimdamas 1 lapą iš Ⓑ krūvelės.

Paaiškinimas

Vienintelis teisingas atsakymas yra C.

Tam, kad laimėtų žaidimą, Emilė ir Justinas mėgins paimti paskutinį lapą. Tam, kad Emilė galėtų paimti paskutinį lapą (arba lapus), turėtų būti likusi tik viena krūvelė. Taip galėtų nutikti tik tuo atveju, jei Justinas paimtų paskutinį lapą iš vienos iš krūvelių prieš pat Emilės ėjimą. Taigi, Emilė turi žaisti taip, kad Justinas neturėtų kito pasirinkimo, kaip tik paimti paskutinį lapą iš vienos iš krūvelių.

A) Jei Emilė paimtų 2 lapus iš Ⓐ krūvelės, Justinas galėtų laimėti žaidimą paimdamas visus 3 lapus iš Ⓑ krūvelės.

B) Jei Emilė paimtų 2 lapus iš Ⓑ krūvelės, Justinas galėtų paimti vieną lapą iš Ⓐ krūvelės. Tokiu būdu Emilė būtų priversta paimti paskutinį lapą iš vienos iš krūvelių ir Justinas kitu ėjimu laimėtų.

C) Jei Justinas paimtų vieną lapą iš Ⓐ krūvelės, Emilė galėtų paimti du lapus iš Ⓑ krūvelės taip priversdama Justiną savo eilės metu paimti paskutinį lapą iš vienos iš krūvelių. Taigi, Emilė laimėtų.

D) Jei Justinas paimtų vieną lapą iš Ⓑ krūvelės, Emilė turėtų paimti vieną lapą iš vienos iš krūvelių. Tačiau tokiu atveju Justinas paimtų vieną lapą iš kitos krūvelės, taigi, kito savo ėjimo metu Emilė turėtų paimti paskutinį lapą iš vienos iš krūvelių ir pralaimėtų. Galime pastebėti, kad Justino pradinis ėjimas šioje situacijoje yra geriausias pradinis ėjimas žaidėjui, kuris pradeda žaidimą.

Matome, kad teisingas atsakymas yra C.

Tai informatika!

Ši užduotis yra gerai žinomo žaidimo „Nim“ variacija. „Nim“ taip pat buvo vienas iš pirmųjų kompiuterizuotų žaidimų. Šiam žaidimui sukurtas kompiuteris „Nimatron“ (1940 m.) laikomas pirmuoju istorijoje žaidimų kompiuteriu! „Nim“, kaip ir „Kryžiukai-nuliukai“, yra vienas iš žaidimų, kuriuos žaidžiant neįmanoma pralaimėti žinant laimėjimo strategiją.

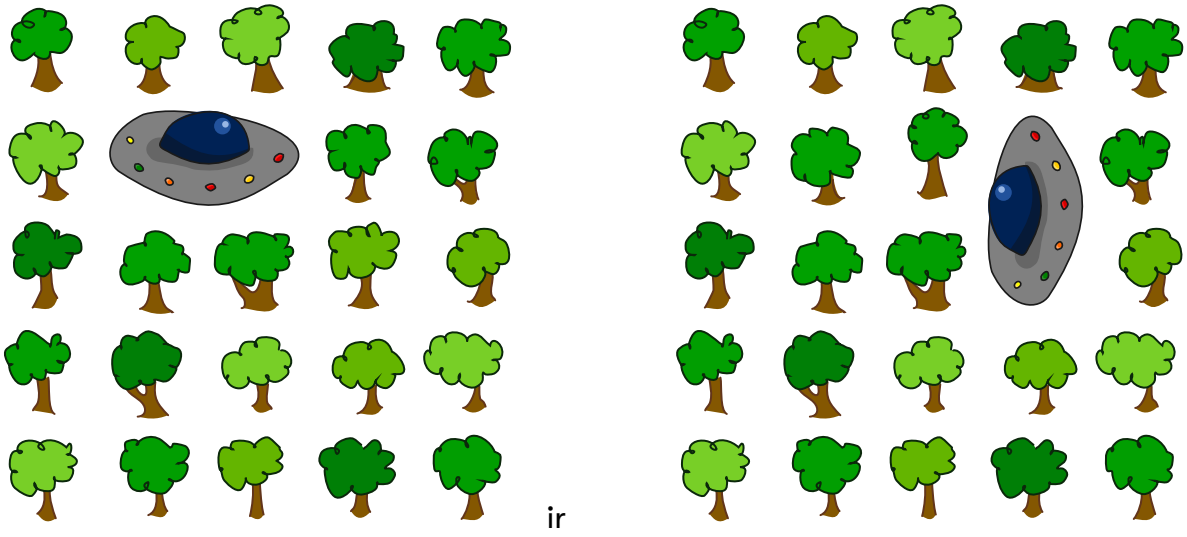
Suprasti šio žaidimo laimėjimo strategiją apgalvojant visas įmanomas situacijas gali būti nelengva. Ypač, jei duotoji situacija yra sudėtingesnė – su daugiau krūvelių ir lapų. Norint supaprastinti tokio uždavinio (ir panašių uždavinių) sprendimą, galima spręsti paprastesnius to paties uždavinio atvejus ir atsakymus vėliau panaudoti sprendžiant vis sudėtingesnius uždavinius. Informatikoje tokia strategija, kai vis sudėtingesniems uždaviniams spręsti panaudojami paprastesnių atvejų atsakymai, vadinama dinaminiu programavimu. Programuotojai dažnai naudoja dinaminį programavimą norėdami padidinti kuriamų programų efektyvumą ir greitį. Šioje užduotyje mes taip pat naudojome panašią strategiją pastebėdami, kurios situacijos veda prie laimėjimo ar pralaimėjimo. Vėliau mums susidūrus su tomis situacijomis nereikėjo vėl atlikti tų pačių veiksmų.

13. Išsaugok medžius!

Bebrų planetoje yra daug gražių medžių, bet nėra vietų, kur galėtų nusileisti kosminiai laivai. Astronautas nori iškirsti medžius, kad galėtų įrengti erdvėlaivių nusileidimo aikšteles. Girininkas nori išsaugoti kuo daugiau medžių. Jie susitaria taip:

- girininkas gali pažymėti tris medžius, kurių negalima nupjauti;
- medį galima nupjauti tik tuo atveju, jei jo reikia nusileidimo aikštelei įrengti (medžio negalima nukirsti be priežasties).

Medžiai išdėstyti 5x5 tinkleliu. Erdvėlaiviai yra gana dideli ir jiems nusileisti reikia dviejų gretimų medžių užimamos erdvės. Erdvėlaivio aikštelei reikalingi nukirsti medžiai pateiktame paveiksle gali būti horizontaliai arba vertikaliai greta, bet ne įstrižai.



Dvi galimos kosminio laivo nusileidimo aikštelės

Kiek nusileidimo aikštelių bus pastatyta, jei tarsime, jog girininkas išmintingai pažymi tris medžius, kad išsaugotų kuo daugiau medžių, ir astronautas išmintingai iškerta medžius, kad pastatytų kuo daugiau nusileidimo aikštelių?

A. 11

B. 10

C. 9

D. 8

Paaiškinimas

Kadangi erdvėlaiviui nusileisti visada reikia dviejų gretimų laukelių, į medžių tinklelį galime žiūrėti kaip į šachmatų lentą:

1	2	3	4	5
10	9	8	7	6
11	12	13	14	15
20	19	18	17	16
21	22	23	24	25

Kosminis laivas visada užima vieną juodą ir vieną baltą kvadratą. Tinklelyje yra lygiai 13 baltų ir 12 juodų kvadratų, todėl jame gali nusileisti ne daugiau kaip 12 erdvėlaivių. Jei baltoje aikštelėje pažymėtumėte vieną medį, bendras skaičius padidėtų iki 12+12 ir vis tiek būtų galima nusileisti ne daugiau kaip 12 erdvėlaivių. Tai lemia, kad medžius reikia žymėti tik juoduose kvadratuose. Jei pažymėsite tris juodus kvadratus, užblokuosite tris galimas nusileidimo aikšteles, todėl nusileidimo aikštelių gali būti ne daugiau kaip 9.

Dabar taip pat turėtume įrodyti, kad trijų juodų kvadratų žymėjimas negali užblokuoti daugiau nei trijų potencialių nusileidimo aikštelių. Tai parodys, kad atsakymas yra lygiai 9. Norėdami tai padaryti, sunumeruojame šachmatų lentos kvadratus, kaip parodyta schemoje. Visada galite nutupdyti erdvėlaivį, kurio viena dalis bus ant lyginio skaičiaus kvadrato su skaičiumi *n*, o kita dalis - kvadrate *n*+1.

Jei „išskleisime“ kvadratus iš lentos ta tvarka, kuria juos sunumeravome, pamatysime šią grandinę:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Medžio žymėjimas juodame kvadrate iš esmės reiškia, kad grandinė perskiriama į dvi dalis toje vietoje, kurioje pažymėjome juodą kvadratą. Jei grandinę su *n* juodų kvadratų perskirsime į dvi dalis, turėsime kairiąją ir dešiniąją pusę. Kiekviena pusė prasideda ir baigiasi baltu kvadratu ir turi pakaitomis baltus ir juodus kvadratus. Bendras juodų kvadratų skaičius dabar yra *n*-1.

Pavyzdžiui, jei pirmiausia pažymėsite 4-ą medį, turėsite dvi 1–3 ir 5–25 grandines. Jei tada pažymėsite 8-ą medį, antrąją grandinę padalysite į 5–7 ir 9–25. Jei po to pažymėsite 10-ą medį, dešinę grandinę padalysite į 9–9 ir 11–25.

Kiekvienoje grandinėje vis dar galima nutupdyti lygiai tiek erdvėlaivių, kiek turime juodų kvadratų.

Tai rodo, kad jei juoduose kvadratuose pažymėsime tris medžius, užblokuosime lygiai 3 nusileidimo aikšteles.

Tai informatika!

Šią užduotį galite pabandyti išspręsti išbandydami visus tris medžių žymėjimo derinius ir tada nustatyti, kiek erdvėlaivių dar galite nutūpti, bet tai užtruktų labai ilgai.

Šioje užduotyje parodoma keletas loginio mąstymo gudrybių, susijusių su abstrakcija. Norint išspręsti šią užduotį, į erdvėlaivius reikia žiūrėti kaip į 2x1 stačiakampius, kuriuos dedate ant lentos, užpildytos kvadratais.

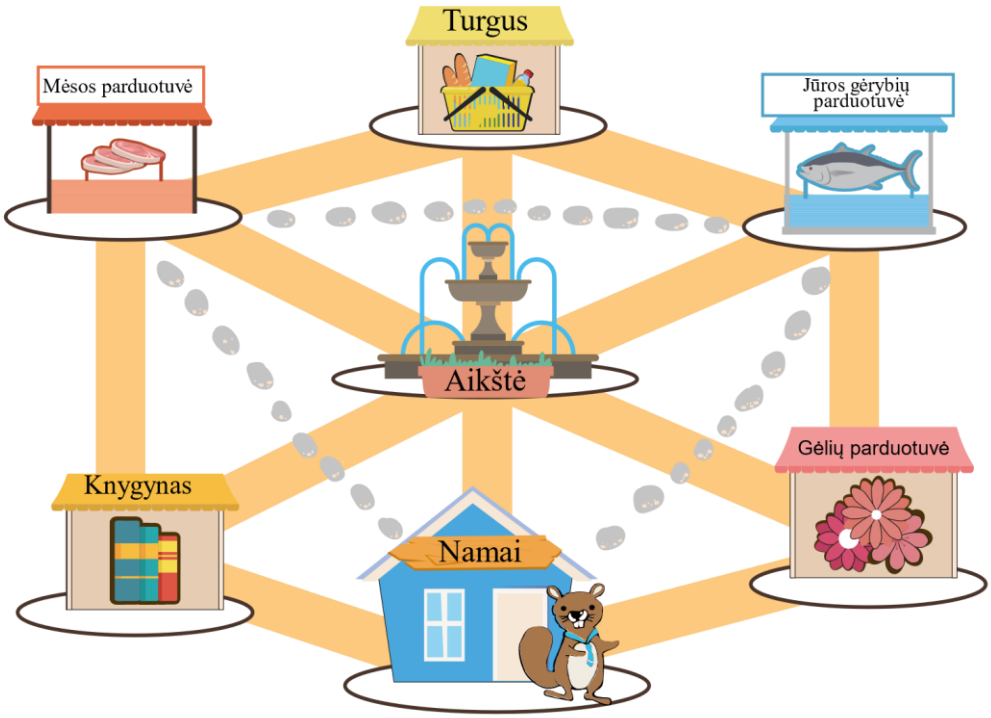
Dažnai šie 2x1 stačiakampiai, kaip ir atitinkamame žaidime, vadinami domino kaladėlėmis.

Jei Jus domina, yra keletas sudėtingesnių algoritmų, kurie parodytų, keliais skirtingais būdais galėtų nusileisti kosminiai laivai (čia buvo klausiama ne to, tačiau informatikams toks klausimas galėtų logiškai iškilti).

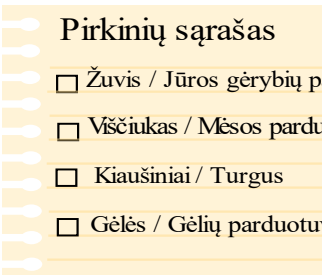
14. Maisto produktų pirkimas

Schemoje pavaizduotas kaimas, kuriame gyvena bebrų šeima. Norėdami pasiekti pastatus, bebrai eina arba pelkėtu , arba akmenuotu taku . Pėsčiomis atstumą tarp bet kurių dviejų pastatų pelkėtu taku Mažasis Bebras įveikia per 5 minutes, o akmenuotu taku – per 8 minutes.

Pavyzdžiui, iš aikštės į gėlių parduotuvę ar knygyną Mažasis Bebras nueina per 5 minutes, o iš jūros gėrybių parduotuvės ar mėsos parduotuvės sugrįžta namo per 8 minutes.



Mama bebrė prašo Mažojo Bebro nupirkti maisto produktų. Pirkinių sąrašas toks:



Mažasis Bebras turi pradėti kelionę iš namų, nupirkti visus produktus ir grįžti namo. Kad žuvis išliktų šviežia, Mažasis Bebras turi apsilankyti jūros gėrybių parduotuvėje prieš pat grįždamas namo.

Kiek mažiausiai laiko minutėmis Mažasis Bebras gali sugaišti išėjęs iš namų, vaikščiodamas po parduotuves ir su pirkiniais grįždamas namo?

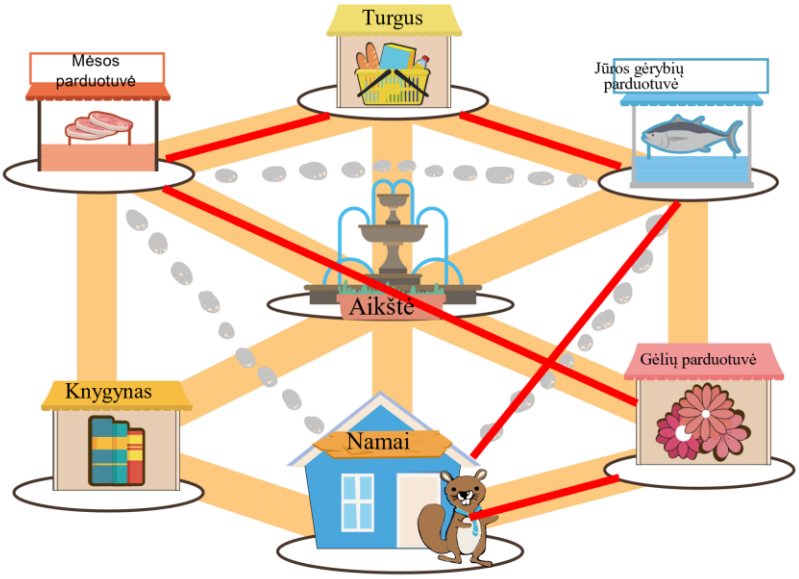
Paaiškinimas

Tarkime, kad esate Mažasis Bebras, kuriam reikia apsipirkti ir grįžti namo. Kad žuvis išliktų šviežia, jūros gėrybių parduotuvė turėtų būti paskutinė parduotuvė prieš grįžtant namo. Todėl visą užduotį galima išskaidyti į dvi atskiras užduotis: (1) apsipirkti mėsos ir gėlių parduotuvėse bei turguje, (2) nusipirkti žuvies ir grįžti namo. Lentelėje pavaizduotos pirkimo galimybės kiekvienam daliniam uždaviniui atlikti:

(1) Pirmieji 3 pirkimai		(2) Paskutinis pirkimas jūros gėrybių parduotuvėje		Visas laikas (min)
Kelias	Mažiausias laikas (min)	Kelias	Mažiausias laikas (min)	
Namai – mėsos parduotuvė – gėlių parduotuvė – turgus	28	Turgus – jūros gėrybių parduotuvė – namai	13	41
Namai – mėsos parduotuvė – turgus – gėlių parduotuvė	23	Gėlių parduotuvė – jūros gėrybių parduotuvė – namai	13	36
Namai – gėlių parduotuvė – turgus – mėsos parduotuvė	20	Mėsos parduotuvė – jūros gėrybių parduotuvė – namai	16	36
Namai – gėlių parduotuvė – mėsos parduotuvė– turgus	20	Turgus – jūros gėrybių parduotuvė – namai	13	33
Namai – turgus – mėsos parduotuvė – gėlių parduotuvė	25	Gėlių parduotuvė – jūros gėrybių parduotuvė – namai	13	38
Namai – turgus – gėlių parduotuvė – mėsos parduotuvė	30	Mėsos parduotuvė – jūros gėrybių parduotuvė – namai	16	46

Ištiriame visus šešis galimus skirtingus būdus. Paskutiniame stulpelyje apskaičiuojamas kelionės laikas visais 6 atvejais.

Maršrutas, kurį turėtų nueiti Mažasis Bebras:



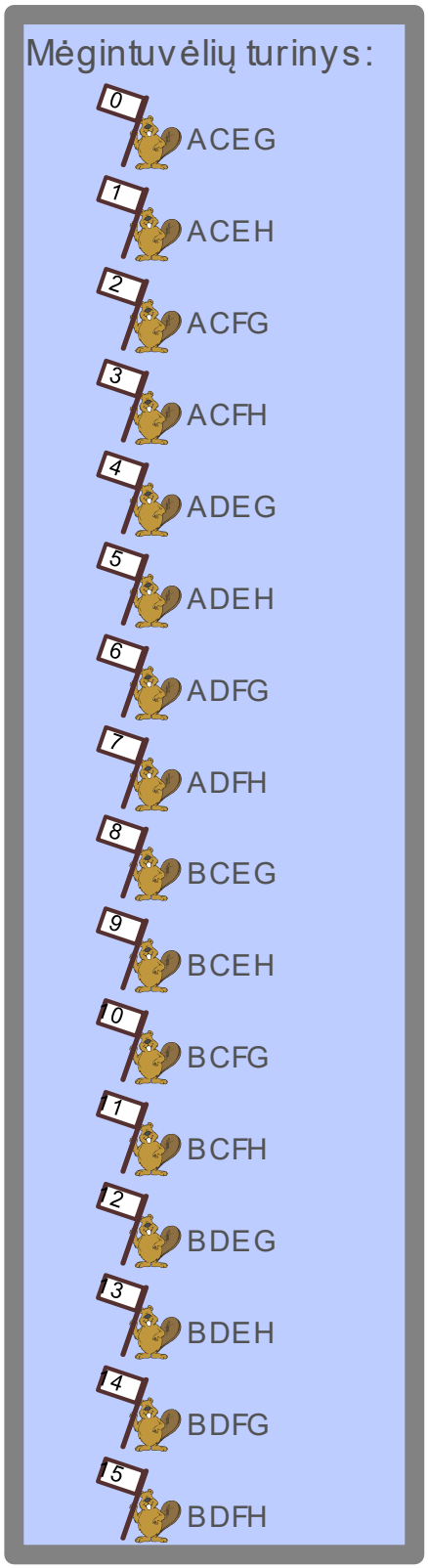
Tai informatika!

Šiam uždaviniui trumpiausią laiką (kelią) galima rasti braižant galimus maršruto planus. Tačiau realiame gyvenime, kai vietoje keliolikos kelių yra tūkstančiai, trumpiausio kelio radimui pasitelkiamos technologijos, pavyzdžiui, navigacijos (GPS) ir maršrutų planavimo programos.

Informatikos moksle grafai yra įprastas būdas parodyti ryšius tarp duomenų. Grafai gali būti naudojami ryšiams tarp objektų vaizduoti. Grafai taip pat padeda lengviau aprašyti ryšius (dažnai vaizduojamus briaunomis, šiuo atveju briaunos atitinka takus) tarp pagrindinių sudėtingų sąvokų (dažnai vaizduojamų viršūnėmis, šiuo atveju tai pastatai).

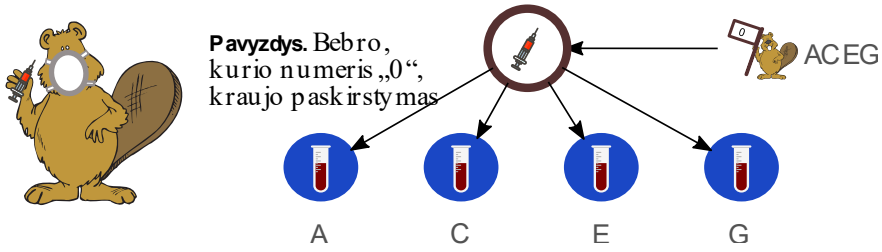
Grafų teorijoje trumpiausio kelio uždaviniu siekiama rasti kelią (briauną), kurio atstumas tarp dviejų objektų (viršūnių) grafe yra mažiausias. Kartais keliai turi skirtingus svorius, todėl atstumą tarp dviejų viršūnių reikia padauginti iš svorių.

15. Kraujo tyrimas



Daktarė Beverli žino, kad serga tik vienas iš 16 jos pacientų bebrų. Ji turi kiekvieno paciento kraujo mėginį, tačiau turi tik 8 mėgintuvėlius A, B, C, D, E, F, G ir H. Sumaišiusi 16 pacientų kraują į 8 mėgintuvėlius, ji žino, kad jai reikia tik 4 kraujo tyrimų, kad išsiaiškintų, kuris iš 16 jos pacientų serga.

Norėdama surasti sergantį bebrą, ji įpila šiek tiek kiekvieno paciento kraujo į 4 skirtingus mėgintuvėlius. Ji kruopščiai laikosi mėgintuvėlių paskirstymo plano. Pavyzdžiui, numeriu „0“ pažymėto bebro kraujas supilamas į A, C, E ir G mėgintuvėlius. Visi paskirstymai pavaizduoti kairėje pateiktame paveiksle.



Daktarė Beverli jau ištyrė 3 mėgintuvėlius ir nustatė, kad A mėgintuvėlis rodo infekciją, o B ir C mėgintuvėliai rodo, kad infekcijos nėra.

Daktarei liko tik vienas tyrimas.

Kurį mėgintuvėlį daktarė Beverli pasirinks sergančiam bebrui identifikuoti?

1) mėgintuvėlį B

2) mėgintuvėlį C

3) mėgintuvėlį F

4) mėgintuvėlį H

Paaiškinimas

Teisingas atsakymas –mėgintuvėlį H.

	A	B	C	D	E	F	G	H
0	X		X		X		X	
1	X		X		X			X
2	X		X			X	X	
3	X		X			X		X
4	X			X	X		X	
5	X			X	X			X
6	X			X		X	X	
7	X			X		X		X
8		X	X		X		X	
9		X	X		X			X
10		X	X			X	X	
11		X	X			X		X
12		X		X	X		X	
13		X		X	X			X
14		X		X		X	X	
15		X		X		X		X

A yra infekcija

C infekcijos nėra

E infekcijos nėra

Serga arba bebras „6“ arba bebras „7“

Lentelėje prie kiekvieno bebro keturi „X“ ženklai rodo, kuriuose keturiuose mėgintuvėliuose yra jo kraujo. Remiantis jau atlikto tyrimo rezultatais, atitinkama spalva pažymėti tam tikri langeliai. Infekciją rodančio A mėgintuvėlio atveju visi 8 pacientai, kurių kraujo yra tame mėgintuvėlyje, yra galimi ligoniai. Kiti aštuoni yra sveiki. Jei mėgintuvėlis rodo, kad infekcijos nėra, visi pacientai, kurių kraujas yra tame mėgintuvėlyje, yra sveiki. Taigi, remiantis jau atliktu tyrimu, serga arba 6-as, arba 7-as pacientas. Žinoma, kad serga tik vienas pacientas, todėl atlikus G arba H tyrimą (nepriklausomai nuo to, ar rezultatas rodys infekciją ar ne) galima nustatyti, kuris iš dviejų pacientų serga. G nėra vienas iš pasirenkamų variantų, todėl teisingas atsakymas turi būti H.

Tai informatika!

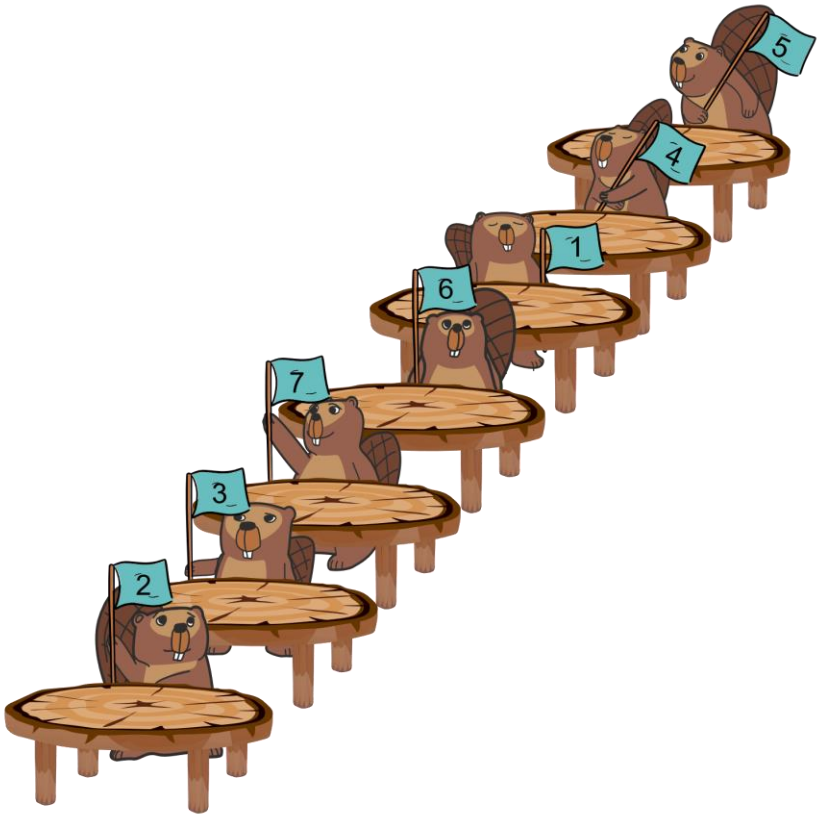
Grupinis testavimas – tai metodas, kurio ištakos siekia Antrojo pasaulinio karo metais Ričardo Dorfmano atliktus kraujo tyrimus. Nuo to laiko šis metodas taikomas daugelyje sričių, ne tik biologijoje, bet ir inžinerijoje bei informatikoje, ypač defektų aptikimo srityje.

Egzistuoja keletas grupinio testavimo algoritmų. Kuris iš jų tinkamiausias, priklauso nuo sprendžiamo uždavinio. Apibendrintas dvejetainio skaidymo algoritmas yra patogus metodas, kuris naudojamas šiame uždavinyje. Mėgintuvėlių pasiskirstymas iš tikrųjų pagrįstas dvejetainiu bebro numerio atvaizdavimu (0 = 00002, 1 = 00012, ..., 15 = 11112), čia A, C, E, G yra visi 0, o B, D, F, H yra 1.

16. Bebrų rikiuotė

Bebrų mokyklos vienoje klasėje yra tik septyni bebrai, kiekvienas iš jų turi po vėliavėlę su numeriu. Jie susodinti iš eilės vienas už kito.

Iš pradžių bebrai sėdi atsitiktine tvarka, kaip parodyta paveikslėlyje.



Klasės mokytoja nori surikiuoti bebrus jų numerių eilės tvarka, kad 1-as numeris būtų priekyje, o 7-as numeris gale. Tą galima padaryti tik atliekant susikeitimo operacijas, kiekvieno susikeitimo metu vietomis gali susikeisti tik du bebrai.

Pavyzdžiui, kai 1-as ir 3-ias bebrai susikeičia vietomis, tai reiškia, kad 3-ias bebras patenka į 1-o vietą, o 1-as – į 3-io vietą.

Naudojant baigtinį susikeitimų skaičių, bebrai bus surikiuoti didėjančia tvarka nuo pirmojo stalo iki tolimiausio.

Koks yra mažiausias susikeitimų skaičius, kad būtų gauta norima bebrų sėdėjimo tvarka?

A) 3 susikeitimai

B) 4 susikeitimai

C) 5 susikeitimai

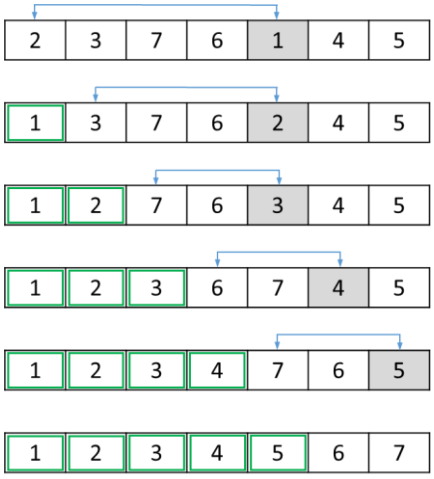
A) 6 susikeitimai

Paaiškinimas

Teisingas atsakymas yra C – 5 susikeitimai, nes turime liepti susikeisti vietomis penkioms skirtingoms bebrų poroms, kaip parodyta paveikslėlyje.

Taikysime rikiavimo išrinkimo metodu algoritmą. Pagal šį algoritmą pradinį sąrašą padalijame į dvi dalis: jau surikiuotų elementų poaibį sąrašo priekyje (kairėje) ir reikiamų surikiuoti elementų poaibį, užimantį likusią sąrašo dalį. Iš pradžių surikiuotų elementų poaibis yra tuščias, o nesurikiuotų poaibį sudaro visas pradinis sąrašas. Algoritmas vykdomas surandant mažiausią (arba didžiausią, priklausomai nuo rikiavimo tvarkos) nesurikiuoto poaibio elementą, sukeičiant jį su kairiausiu nesurikiuotu elementu (jį sutvarkant) ir poaibio ribas perkeliant vienu elementu į dešinę.

Pirmame žingsnyje (kadangi uždavinio sąlygoje nurodyta rikiuoti didėjimo tvarka) pasirenkame mažiausią skaičių (skaičius 1) ir sukeičiame jį su sąrašo pirmosios vietos skaičiumi (skaičius 2). Kitame žingsnyje, pradėdami nuo antrosios sąrašo vietos, pasirenkame mažiausią skaičių (t. y. skaičių 2) ir sukeičiame jį su antrosios sąrašo vietos skaičiumi (skaičiumi 3). Galiausiai mums reikia penkių žingsnių.



Tai informatika!

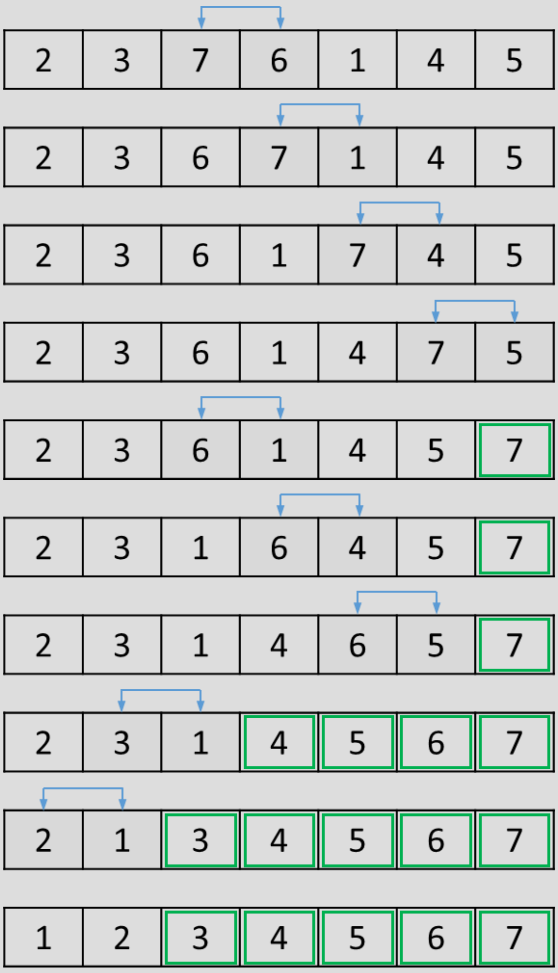
Rikiavimo algoritmai leidžia, kaip rodo jų pavadinimas, surikiuoti informaciją specialiu būdu pagal rikiavimo kriterijų.

Informatikos moksle duomenų rikiavimas yra svarbus tiek kaip atskiras uždavinys, tiek ir kaip kitų daug sudėtingesnių procedūrų dalis. Šioje srityje yra sukurta daug metodų, kurių kiekvienas pasižymi specifinėmis savybėmis ir turi pranašumų bei trūkumų kitų atžvilgiu.

Rikiavimo išrinkimo metodu algoritmas yra tobulesnis nei burbulo metodo algoritmas, nes kiekvieno perėjimo per sąrašą metu atliekamas vienintelis sukeitimas. Tai pasiekiama, einant per sąrašą ir išrenkant mažiausią reikšmę, o baigus sąrašo peržiūrą tą reikšmę perkeliant į reikiamą vietą. Kaip ir rikiavimo burbulo metodu atveju, po pirmosios sąrašo peržiūros mažiausias elementas atsiranda tinkamoje vietoje. Po antros peržiūros kitas mažiausias elementas atsiranda savo vietoje. Šis procesas tęsiamas toliau ir prireikia (n-1) sąrašo peržiūrų n elementams surikiuoti, taigi paskutinis elementas turi būti savo vietoje po (n-1)-osios peržiūros.

Dėl sąrašo peržiūrų skaičiaus sumažinimo rikiavimas išrinkimo metodu paprastai vyksta greičiau nei burbulo metodu, nors daliai rikiavimo uždavinių tai gali būti ir ne pats greičiausias algoritmas (yra daug algoritmų, skirtų „daiktams“ rikiuoti).

Jei palygintume rikiavimą išrinkimo metodu ir burbulo metodu, pamatytume, kad pirmasis yra greitesnis.

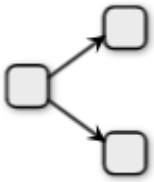


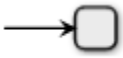
17. Gamybos vienetai

Gamykla gamina kelių rūšių produktus. Kiekvienam produktui yra keli gamybos proceso etapai. Šie gamybos etapai vyksta skirtinguose padaliniuose (▢)₁. Visi gamybos procesai gamykloje yra apibendrinti diagramoje, kurioje kvadratai žymi padalinius, o rodyklė ( ) – gaminio srautą iš vieno padalinio į kitą.

Yra dviejų tipų padaliniai: gamybos ir pakavimo skyriai.

- Iš kiekvieno gamybos padalinio produktai gali būti siunčiami daugiausia į du kitus padalinius.



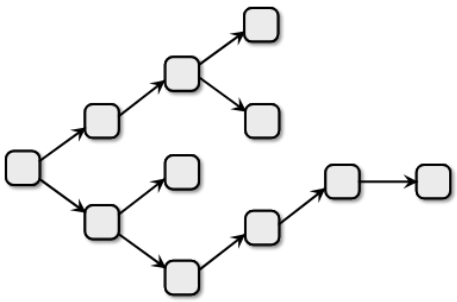
- Iš pakavimo padalinio gaminiai išsiunčiami iš gamyklos. Šis padalinys rodomas be išeinančios rodyklės, o tik su priėmimo rodykle: 

Gamybos proceso sąlygos:

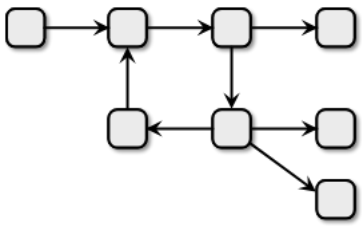
1. *Pirmas padalinys* yra skyrius, kuriame prasideda kiekvieno produkto gamyba.
2. Kiekvienas padalinys turi gauti produktus tik iš vieno kito padalinio.

Kurioje iš šių diagramų teisingai pavaizduoti gamyklos gamybos procesai?

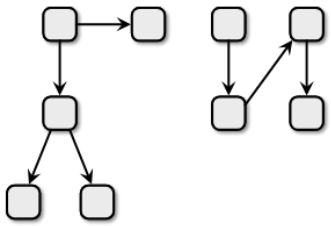
A.



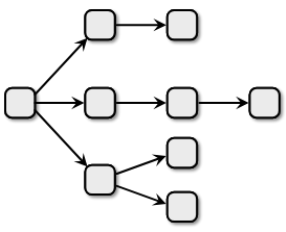
B.



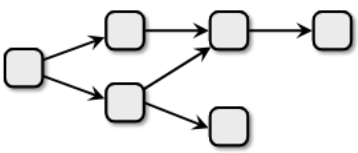
C.



D.

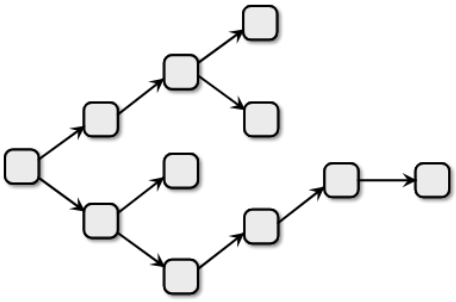


E.



Paaiškinimas

Teisingas atsakymas:



Kairysis kvadratas gali reikšti pirmą padalinį; kiekvienas padalinys gauna produktus tik iš vieno padalinio; pakavimo padaliniai yra be išeinančių rodyklių; kiti padaliniai siunčia produktus daugiausia dviem padaliniams.

B diagrama neteisinga, nes yra kvadratų, kurie gauna produktus iš daugiau nei vieno kvadrato. Tai reiškia, kad atitinkami padaliniai gautų gaminius iš daugiau nei vieno padalinio, o tai neatitinka gamyklos gamybos proceso sąlygų. Iš tiesų, diagramoje yra ciklas, jungiantis 4 padalinius.

C variante yra dvi diagramos, taigi, ir du pirmieji padaliniai, t. y., vaizduojama susivienijimas, susidedantis iš dviejų gamyklų (ar vienos gamyklos du filialai).

D diagrama neteisinga, nes yra kvadratų su daugiau nei 2 išeinančiomis rodyklėmis. Tai reiškia, kad yra padalinių, kurie siunčia produktus daugiau nei dviem padaliniams, o taip negali būti.

C diagrama nėra teisinga, nes, kaip ir B diagramoje, yra kvadratas, į kurį veda rodyklės iš daugiau nei vieno kito kvadrato (t. y., atitinkamas padalinys gautų produktus iš daugiau nei vieno padalinio, o tai neatitinka gamybos sąlygų).

Tai informatika!

Šioje užduotyje pavaizduotos diagramos informatikoje vadinamos grafais. Tai duomenų struktūra, kurią galima naudoti dvejetainiam ryšiui tarp objektų pavaizduoti. Šiuo atveju subjektai yra gamybos padaliniai ir du padaliniai yra susiję: vienas padalinys gali siųsti produktus kitam padaliniui. Grafai dažnai naudojami kompiuterių moksle įvairioms situacijoms modeliuoti; pavyzdžiui, tinklai paprastai modeliuojami grafais. Realios situacijos modeliavimas naudojant tokias abstrakcijas kaip grafus, leidžia informatikams pritaikyti grafų savybes ir grafų algoritmus daugeliui skirtingų realaus gyvenimo uždavinių.

Gamybos proceso savybės lemia grafo savybes. Šiuo atveju grafas yra medis, beje, jis yra susietasis ir neturi ciklų. Medžiai gali būti naudojami modeliuojant hierarchinius santykius, pavyzdžiui, šeimos medyje arba organizacijos schemoje.

18. Matematinė mašina

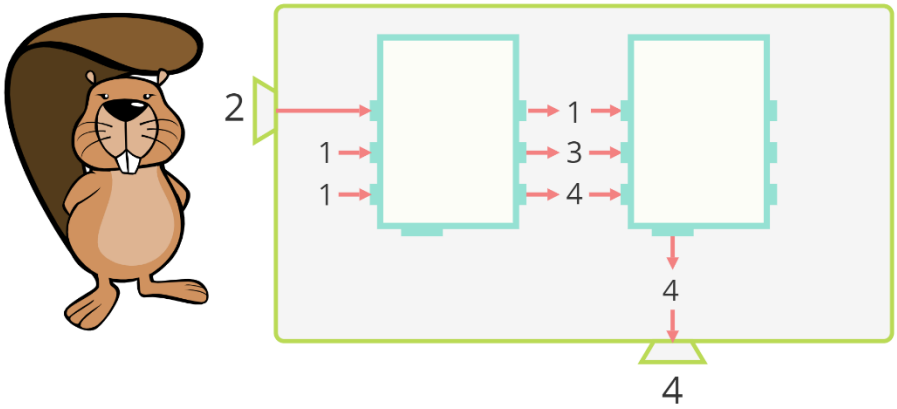
Bebrai sukūrė matematinę mašiną. Ji priima natūraliuosius skaičius ir juos pertvarkiusi pateikia rezultatus. Mašinos viduje naudojami komponentai, kurie visi veikia vienodai. Kiekvienas komponentas priima po tris skaičius ir atlieka tokius veiksmus:

Jei pirmas skaičius yra 1, tai trečias skaičius yra mašinos pateikiamas galutinis rezultatas.

Priešingu atveju:

- Pirmasis skaičius vienetu sumažinamas ir laikomas nauju pirmuoju skaičiumi.
- Antrasis skaičius padidinamas 2 ir laikomas nauju antruoju skaičiumi.
- Naujas antrasis skaičius sudedamas su trečiuoju ir ši suma laikoma nauju trečiuoju skaičiumi.
- Visi trys naujieji skaičiai siunčiami į naują komponentą tokia pačia tvarka.

Paveiksle pavaizduota, kaip matematinė mašina pertvarko skaičių 2, šiuo atveju naudodama tik du komponentus.



Kai tik mašina gauna skaičių, ji siunčia šį skaičių komponentui kaip pirmą skaičių, o likusius du skaičius komponentui pateikia vienetus. Galutiniu mašinos rezultatu laikomas bet kurio komponento pateiktas galutinis rezultatas.

Kurie iš šių skaičių gali būti matematinės mašinos galutiniai rezultatai?

A. 1

B. 50

C. 100

D. 250

E. 400

Paaiškinimas

Teisingi atsakymai yra 1, 100 ir 400.

Mašina, gavusi pradinį duomenį n, sudeda pirmus n nelyginius skaičius, pradėdama 1, ir šią sumą pateikia kaip galutinį rezultatą, todėl šis rezultatas yra n^2.

Tai informatika!

Programavime dažnai didelis uždavinys suskaidomas į mažesnes dalis, sprendžiančias tarpusavyje panašius uždavinius. Šis programavimo būdas vadinamas „skaldyk ir valdyk“ (angl. *divide and conquer*).

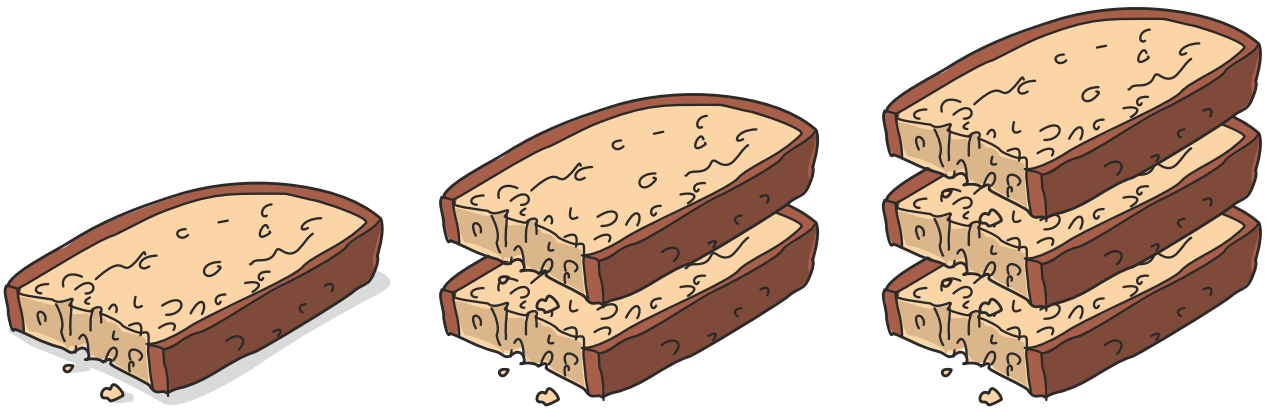
Rekursija yra viena iš pagrindinių informatikos mokslo idėjų. Rekursija naudojama, kai funkcijos ar procedūros kreipiasi pačios į save.

Rekursijos galia akivaizdžiai slypi galimybėje apibrėžti begalinę objektų aibę baigtiniu teiginiu. Taip begalinis veiksmų skaičius gali būti aprašomas baigtine rekursine programa, net jei šioje programoje nėra aiškių pasikartojimų. Vadinamoji uodegos rekursija (angl. *tail call*) yra ypač naudinga ir ją lengva įgyvendinti. Uodegos rekursija gali būti vykdoma nepridėjus naujo programos dėklo kadro (angl. *stack frame*) į dėklą (angl. *call stack*).

19. Tylusis kalbėjimas

Du broliai gyvena vienuolyne. Jie bando laikytis tylos, tačiau ir tyloje norėtų bendrauti. Broliai sugalvojo „kalbėjimosi“ būdą naudodami per pietus gaunamas 6 duonos riekes. Kiekvienas ant stalo sudėlioja savo duoną krūvelėmis nuo 1 iki 6 riekių. Visada naudojamos visos 6 riekės, o jų išdėstymas atitinka žodį, kuri norima „pasakyti“.

Pavyzdžiui, žodis „labas“ galėtų būti perduotas išdėstant duonos gabaliukus taip:



Kiek skirtingų žodžių broliai gali perduoti naudodami tokią sistemą?

Paaiškinimas

Kiekvienas brolis turi 6 duonos riekes. Įsivaizduokime tokį jų dėstymo būdą:

Ant stalo padėti pirmą riekę.

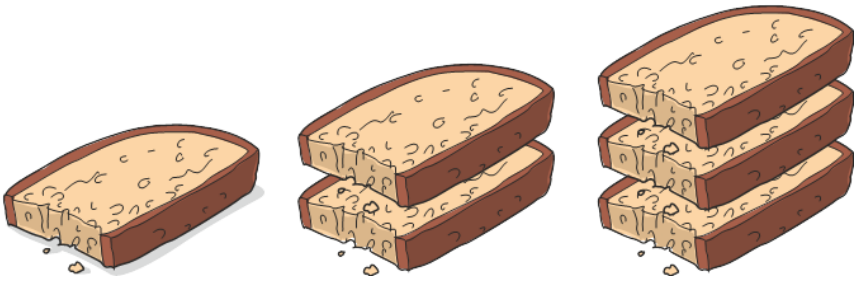
Mesti monetą.

Jei atsiverčia herbas, ant viršaus padėti kitą riekę.

Jei atsiverčia skaičius, pradėti kitą krūvelę.

Monetą mesti, kol visos riekės bus išdėstytos.

Pavyzdžiui, pranešimas



yra rezultatas tokių monetos metimų: skaičius, herbas, skaičius, herbas, herbas.

Kadangi visoms riekėms išdėstyti metama moneta ir kiekvienas metimas turi du variantus, iš viso yra $2^5 = 32$ skirtingų duonos riekių išdėstymo būdų.

Tai informatika!

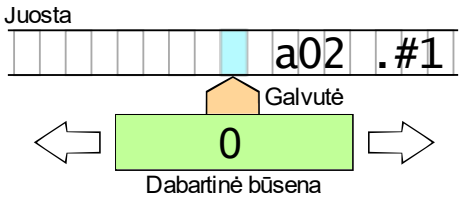
Broliai rado būdą užkoduoti žodžius, kuriais jie nori bendrauti. Kodavimas yra veiksmas, kuriuo ženklui arba kitokiam objektui suteikiamas kodas. Taip duomenys pateikiami kitu (ekvivalenčiu) būdu. Broliai naudoja kodą žodžiams pateikti duonos riekių krūvelėmis.

Taigi pranešimas

būtų dvejetainė seka 10100.

Kodavimas labai įvairiai naudojamas informatikoje. Duomenų kodavimas gali būti taikomas pranešimams įslaptinti. Toks kodavimas vadinamas šifravimu, o šifravimo ir iššifravimo būdų ir priemonių visuma – kriptografija. Kodavimas gali būti naudojamas tam, kad duomenys užimtų mažiau vietos atmintyje – toks kodavimas vadinamas glaudinimu.

20. Tiuringo mašinos



Tiuringo mašina yra abstraktus kompiuterio modelis. Ją sudaro **būsena** ir skaitymo-rašymo **galvutė**, slankiojanti **juosta**, kurioje yra simbolių. Galvutė gali judėti į kairę (**k**) arba į dešinę (**d**) per vieną simbolį vienu metu. Mašina visada pradeda būsena „0“ – tai **pradinė būsena**.

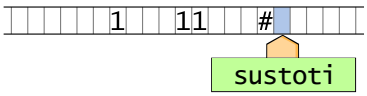
Mūsų Tiuringo mašinos versija gali vykdyti programas, o kiekvieną tokių programų eilutę sudaro 5 elementai, kaip parodyta pateiktoje programoje:

Dabartinė būsena	Dabartinis simbolis	Naujas simbolis	Kryptis	Nauja būsena
0	_	_	d	0
0	1	1	d	1
0	0	_	d	0
0	#	#	d	sustoti
1	0	0	d	1
1	1	1	d	1
1	_	_	d	0
1	#	#	d	sustoti

Pirmoji (nuo viršaus) eilutė, kurioje aprašyta dabartinė būsena ir galvute rodomas dabartinis simbolis, vykdant programą nurodo **naują simbolį**, kuris bus užrašomas dabartinėje juostos pozicijoje, galvutės judėjimo **kryptį** ir naują mašinos **būseną**.

Simbolis „_“ reiškia tarpą. Būsena „sustoti“ nutraukia programos vykdymą. Tuščios eilutės naudojamos tik formatavimui.

Pateiktoji programa pašalina priekinius nulius dvejetainiuose skaičiuose ir pasiekia štai tokią galutinę būseną:



Iš pradinės būsenos galvutė juda į dešinę, kol pasiekia pirmąjį 0, likdama būsenoje 0. Toliau atsižvelgiant į 3-iąją taisyklę (trečioji eilutė), pirmasis 0 pakeičiamas tarpu, judama į dešinę, būsena lieka 0. Tada galvutė skaito „1“, jis paliekamas ant juostos, judame į dešinę ir pereiname į būseną 1. Toliau yra tarpo simbolis, todėl pagal programą paliekamas tarpas juostoje ir pereinama atgal į būseną 0. Taip procesas tęsiamas tol, kol pasiekiame ankščiau pavaizduotą galutinę būseną.

Ką atlieka ši programa?

Dabartinė būsena	Dabartinis simbolis	Naujas simbolis	Kryptis	Nauja būsena
0	1	_	d	1
0	*	*	d	0
1	1	_	d	2
1	*	f	d	sustoti
2	1	_	d	1
2	*	t	d	sustoti

Pastabos:

„*“ 2-ame stulpelyje reiškia „bet kuris simbolis“.

„*“ 3-iame stulpelyje reiškia „tas pats simbolis, kuris buvo perskaitytas“.

A) Programa pakeičia kiekvieną „1“ į „2“.

B) Programa pakeičia kiekvieną „2“ į „1“.

C) Kai programa randa grupę vienetų („1“), ji spausdina „t“ (angl. *true*), jei vienetų skaičius yra lyginis. Priešingu atveju spausdina „f“ (angl. *false*).

D) Kai programa randa grupę vienetų („1“), ji spausdina „t“ (angl. *true*), jei vienetų skaičius yra nelyginis. Priešingu atveju spausdina „f“ (angl. *false*).

Paaiškinimas

Teisingas atsakymas – C.

Esant pradinei būsenai eilutė [0 * * r 0] reiškia, kad praleidžiami visi simboliai, išskyrus vienetus („1“). Jei aptinkamas simbolis „1“, pirmoji eilutė pereina į būseną „1“.

Būsena „1“ reiškia, kad iki šiol aptiktas nelyginis vienetų („1“) skaičius. Jei aptinkamas dar vienas „1“ simbolis, tai mašina pereina į būseną „2“. Jei aptinkamas bet koks kitas simbolis, ne „1“, tai mašina spausdina „t“ ir sustoja.

Būsena „2“ reiškia, kad iki šiol buvo aptiktas lyginis vienetų („1“) skaičius. Jei aptinkamas dar vienas „1“, tai mašina pereina į būseną „1“. Jei aptinkamas bet koks kitas simbolis, ne „1“, tai mašina spausdina „t“ ir sustoja.

Būtent, toks rezultatas gaunamas pasirinkus C variantą.

A) variantas

Jei programa visus vienetus („1“) pakeičia dvejetais („2“), tai programoje turi būti eilutė, kurios antrame stulpelyje būtų „1“, o trečiajame stulpelyje būtų „2“. Tačiau taip nėra. Taigi A variantas netinkamas.

B) variantas

Panašiai kaip A atveju, norėdami pakeisti visus dvejetus („2“) vienetais („1“), programos eilutės antrame stulpelyje (dabartinis simbolis) turi būti „2“ arba „*“ ir „1“ trečiame stulpelyje (naujas simbolis). Tačiau tokios eilutės nėra. Taigi B yra neteisingas.

D) variantas

Kadangi radome teisingą C variantą, tai D variantas turi būti neteisingas. Tiksliau: lyginis vienetų („1“) skaičius keičia būseną į „1“ (arba iš būsenos „0“, kai perskaitomas „1“, arba iš būsenos „2“, kai perskaitomas dar vienas „1“). Bet kuriuo iš šių atvejų, esant būsenai „1“ ir lyginiam vienetų („1“) skaičiui, toliau galima arba:

a) skaityti kitą „1“ ir gauti būseną „2“, tačiau tuo pačiu gautume lyginį vienetų skaičių, arba

b) skaityti kitokį simbolį, kuris nutrauktų programą ir išspausdintų „f“ (false).

Taigi D variantas neteisingas.

Tai informatika!

Tiuringo mašina yra abstraktus skaičiuojančiojo kompiuterio modelis, kurį 1936 m. sukūrė britų matematikas Alanas Tiuringas. Nors Tiuringo mašina yra paprastas konceptas, mokslininkai sutaria, kad bet koks algoritmas, kuris veikia klasikiniame kompiuteryje, turi veikti ir taikant Tiuringo mašiną, nors ir nelabai efektyviai. Ir atvirkščiai, jei galima įrodyti, kad algoritmas negali būti įvykdytas Tiuringo mašina, tai tas algoritmas negali būti atliktas ir klasikiniu kompiuteriu. Taigi Tiuringo mašiną galime laikyti kompiuteriu, kuris kiek įmanoma supaprastintas, kad būtų galima daryti išvadas ir remtis bendrosiomis klasikinių kompiuterių savybėmis. Kvantiniams kompiuteriams taikoma išimtis, jie nepatenka į tą pačią kategoriją su klasikiniais kompiuteriais.

Atvėrę http://morphett.info/turing tinklalapį galite rasti Tiuringo mašinos simuliatorių su integruotomis programomis, kurias galima įkelti ir paleisti vykdyti. Pateiktoji programa daugina dvejetainį skaičių iš 2 (pridėdama jį prie savęs):

```
; -------------------------------------------------------------
; http://morphett.info/turing
; Padauginti dvejetainį skaičių iš 2
; Įvesti: vienas dvejetainis skaičius, pvz.:  '10011110' (158), 1111 (15)
; Išvestis:              pvz. '100111100' (316), 11110 (30)
;<dabartinė būsena> <dabartinis simbolis> <naujas simbolis> <kryptis> <nauja būena>'
; go to end
0 1 1 r 0
0 0 0 r 0
0 _ _ l 1
; tęsti be perkėlimo
1 0 0 l 1
1 1 0 l 2
; tęsti perkėlus
2 0 1 l 1
2 1 1 l 2
2 _ 1 * sustoti
```

21. Suoliukų dirbtuvės

Bebras Albertas gamina suoliukus. Kiekvienas suoliukas turi turėti keturias vienodo ilgio kojas, kurios lemia suolo aukštį. Kadangi klientų pageidavimai gali skirtis, kiekvienas suoliukas gali būti skirtingo aukščio.

Albertas medžiagos suoliukų kojoms ieško miške, todėl ne visada pavyksta rasti tokio pat ilgio kojas. Tačiau Albertas gali sutrumpinti kiekvieną koją iki bet kokio ilgio (sutrumpinus koją, likusios dalies naudoti negalima).

Šiuo metu sandėlyje Albertas turi iš viso 32 tokias kojas:

Ilgis	10	9	8	7	6	5	4	3	2
Kiekis	3	6	3	3	5	3	3	2	4

Kojų trumpinimas yra labai sunkus procesas ir reikalauja daug pastangų. Taigi, Albertas nori jūsų pagalbos, kad galėtų sužinoti **kokį minimalų kojų skaičių jis turi sutrumpinti, kad padarytų aštuonis (8) suolus?** Suolų aukštis gali būti skirtingas.

Paaiškinimas

Teisingas atsakymas: 6.

Yra dvi nelabai akivaizdžios taisyklės, kaip rasti optimalų sprendimą tuo atveju, kai reikia naudoti visas kojas.

1 taisyklė: jei kažkurio ilgio kojų kiekis didesnis nei 4, naudokite kuo daugiau galimų 4 kojų rinkinių.

Jei sutrumpinsime 4 kojas, nepaisant galimybės suoliuką sukurti netrumpinant, tai pabaigoje iš šių 4 kojų gausite vieną komplektą suoliukui, bet papildomai turėsite dar 4 patrumpinimus. Taigi per didelis 4 kojų rinkinių trumpinimas negali būti optimalus. Taikant 1 taisyklę pateiktiems duomenims, naudokite visus vienodo ilgio 4 kojų rinkinius (kaip matome, tinka ilgiams 9, 6 ir 2). Kai atimame visus tuos rinkinius, lieka tokia kojų lentelė:

Ilgis	10	9	8	7	6	5	4	3
Kiekis	3	2	3	3	1	3	3	2

2 taisyklė: Visos vienodo ilgio kojos naudojamos suoliuko gamybai arba trumpinamos. Ši taisyklė akivaizdi ilgiausioms ir trumpiausioms kojoms. Mūsų atveju 10 ilgio kojos turėtų būti sutrumpintos, nes nėra kojų, kurias būtų galima sutrumpinti iki tokio ilgio. Suoliuko gamybai bus naudojamos visos 3 ilgio kojos, nes nėra jokios priežasties jas trumpinti. Pažiūrėkime, kiek papildomų kojų reikia kiekvienam ilgiui:

Ilgis	10	9	8	7	6	5	4	3
Kiekis	3	2	3	3	1	3	3	2
Kiek trūksta kojų	0	2	1	1	3	1	1	2

Reikėtų sutrumpinti bent trijų ilgių kojas – sutrumpinus vos dviejų ilgių kojas neįmanoma gauti reikiamo bendro skaičiaus kojų, kurių trūksta iki pilno rinkinio (3+3 < 2+1+3+1+1+2).

Mažiausias kojų skaičius, kurį sudaro trijų skirtingų ilgių kojos yra 6 (ilgiai 10, 9 ir 6). Sutrumpinus 6 ilgio koją iki 5 ilgio, gaunamas visas 4 kojų, kurių ilgis 5, komplektas. Naudojant penkias ilgiausias kojas, kurių ilgis 10 ir 9, galima sukomplektuoti tokius 4 kojų komplektus: 8, 7, 4 ir 3. Taigi optimalus sprendimas – 6 kojos turėtų būti sutrumpintos.

Tai informatika!

Optimalus sprendimas – tai įmanomas sprendimas, kai funkcija pasiekia didžiausią (arba mažiausią) reikšmę – pavyzdžiui, didžiausią pelną arba mažiausią kainą. Globalus optimalus sprendimas yra tas, kuriame nėra kitų įmanomų sprendimų su geresnėmis funkcijos reikšmėmis. Lokaliai optimalus sprendimas yra toks, kai „greta“ nėra kitų įmanomų sprendimų su geresnėmis funkcijų reikšmėmis – galite tai įsivaizduoti kaip tašką „viršūnės“ viršuje arba „slėnio“ apačioje, kuris gali būti sudaromas iš funkcijos ir (arba) apribojimų.

Dinaminis programavimas yra algoritmavimo metodas, skirtas optimizavimo uždaviniui išspręsti, suskaidant jį į paprastesnes dalis ir pasinaudojus tuo, kad optimalus viso uždavinio sprendimas priklauso nuo optimalaus jo sudedamųjų uždavinių sprendimo. Optimizavimas pritaikomas daugelyje informatikos sričių, todėl tai yra labai svarbi sąvoka.

Ši užduotis yra puikus pavyzdys mokiniams, nes parodo, jog optimalus sprendimas ne visada yra akivaizdus ir jo negalima rasti naudojant kokį nors godųjį (angl. *greedy*) algoritmą, pabrėžiantį algoritmų ir programavimo svarbą.

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“ uždavinių rinkinys tinka Mokyklos pedagogikos studijų programos moduliui „Informatikos didaktika“. Studentai, būsimi mokytojai, nagrinėdami „Bebro“ uždavinius susipažįsta su įvairiais informatikos konceptais, gilinasi į sudėtingesnius informatikos konceptus ir išmoksta juos paaiškinti.