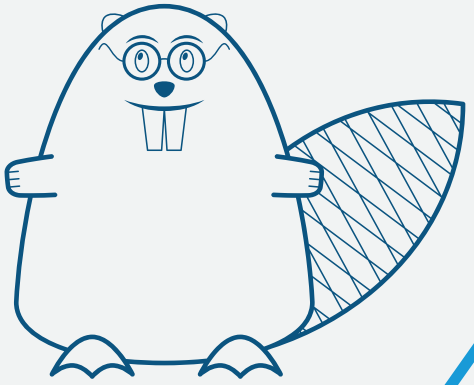
VISMA

Bebr[a]s

2019. gada
1. kārtas uzdevumi ar atbildēm
11.-12. klasei



SATURS

Senā valoda

Kanāda

Pētnieki uz alas sienas atklāja piecus senus vārdus:

paqrooob
puue
t'seqrub
meoub
lai'laiqy

Lai noteiktu, pie kādas valodas pieder katrs no vārdiem, tiek izmantota punktu sistēma.
Katram vārdam sākotnēji tiek piešķirti 10 punkti.
Tad punktu skaits var tikt mainīts, izmantojot šādus nosacījumus:

Sākas ar burtu “p”	-2
Beidzas ar burtu “b”	-2
Satur vairāk par 6 rakstzīmēm	+3
Satur burtu “q”, aiz kura uzreiz seko burts “r” vai burts “y”	-4
V;ardā ir trīs patskaņi (burti “a”,”e”,”i”,”o” vai “u”) pēc kārtas	+5
Satur rakstzīmi - apostrofu (')	+1

Ja rezultātā vārdam ir 10 vai vairāk punktu, sistēma nosaka, ka vārds pieder pie seno bebriešu valodas. Ja tā nav - pie koknešu valodas.

Piemēram, vārdam palliob ir deviņi punkti: 10-2-2+3=9. Tātad vārds pieder pie koknešu valodas.

Uzdevums:

Cik daudz vārdu uz alas sienas pieder koknešu valodai? (Atbildi rakstīt tikai ar skaitļiem, bez nekādām citām rakstu zīmēm)

Pareizā atbilde: 1

Tabula zemāk apkopo iegūtos punktus katram vārdam:

Vārds	Punkti	Valoda
paqrooob	10-2-2+3-4+5=10	Seno bebriešu
puue	10-2+5=13	Seno bebriešu
t'seqrub	10-2+3-4+1=8	Koknešu
meoub	10-2+5=13	Seno bebriešu
lai'laiqy	10+3-4+1=10	Seno bebriešu

Kā redzams - tikai viens vārds (t'seqrub) pieder koknešu valodai.

Kā tas ir saistīts ar informātiku?

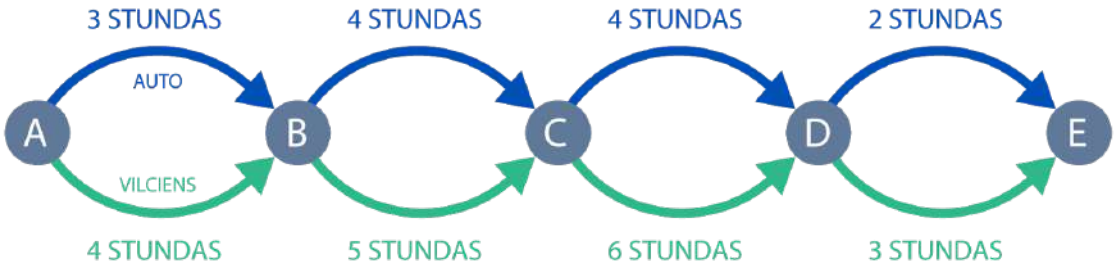
Datorzinātņu pielietošana dabīgās valodas pētniecībā - datorlingvistika ir aizraujoša zinātnes nozare, kurā joprojām noris aktīva pētniecība. Viena no pētījumu jomām ir automātiska teksta pārvēršana balss ziņojumos vai otrādi. Daudzi no mums šo tehnoloģiju jau izmanto ikdienā lietojot viedierīces vai mājas elektroniku. Otrs raksturīgs piemērs ir teksta tulkošana no vienas valodas citā. Ja valoda, no kuras nepieciešams tulkot, nav zināma, tad pirmais tulkošanas solis ir noskaidrot, kas tā par valodu. Un tieši šī uzdevuma risinājums vienkāršodā veidā ir parādīts dotajā uzdevumā.

Dabīgās valodas uzdevumu risināšana vispārīgā gadījumā ir ļoti grūtas un to perfekti atrisinājumi joprojām nav atrasti. Šajā uzdevumā tiek lietotas heiristikas, kas ar vienkāršu likumu palīdzību ļauj analizēt vārdus, cerībā, ka lielā skaitā gadījumu šī metode dos derīgas atbildes. Dažkārt pat vienkāršu heiristiku lietošana var dot apbrīnojami labus rezultātus. Psihologi strādā pie programmām, kas ļautu noteikt teksta rakstītāja vecumu un dzimumu, kā arī noteikt, ja kāds melo.

Saudzēsim vidi

Holande

Bebram Jānim ir jāceļo no pilsētas A uz pilsētu E caur B, C un D. Ceļojot no vienas pilsētas uz otru, viņš var izmantot vilcienu. Vilciena braukšanas ilgums ir redzams attēlā zemāk. Vilciena vietā viņš katrā pilsētā var īrēt mašīnu. Braukšanas ilgums redzams attēla augšpusē.



Jāni uztrauc vide, tāpēc viņš vēlas ceļot ar mašīnu pēc iespējas mazāk. Tā kā viņam galapunktā E ir jāierodas 15 stundu laikā, tad viņš ir spiests vismaz daļu attāluma ceļot ar mašīnu.

Uzdevums:

Kāds ir mazākais laiks, kuru Jānis var pavadīt, ceļojot ar mašīnu, lai galamērķī nonāktu laikā?

Pareizā atbilde: 6 stundas

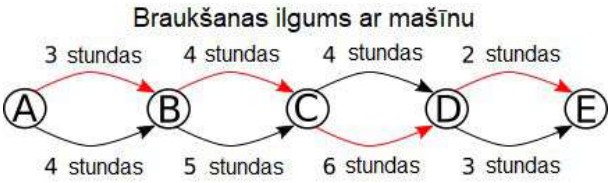
Jānis dod priekšroku ceļošanai ar vilcienu. Ja viņš visu ceļojumu veiktu ar vilcienu, tas aizņemtu 18 stundas. Tāpēc viņam ir jāaizvieto vismaz 3 stundas vilciena brauciena ar braucienu mašīnā. Mēs varam saskaitīt visas iespējas, lai to būtu iespējams izdarīt.

	Ietaupītais laiks, braucot ar mašīnu vilciena vietā	Brauciena ilgums
No A uz B	1 stunda	3 stundas
No B uz C	1 stunda	4 stundas
No C uz D	2 stundas	4 stundas
No D uz E	1 stunda	2 stundas

Ir jāsamazina kopējais braukšanas ilgums tā, lai ietaupītas sanāktu vismaz trīs stundas. To ir iespējams veikt tikai ar pārsēšanos.

Ir iespējami četri risinājumi. Visos gadījumos ceļā jāpavada 15 stundas, lai nokļūtu no punkta A līdz punktam E. Ir jāizvēlas viens no braukšanas veidiem, kurā vismazāk laika tiek pavadīts braucot ar mašīnu.

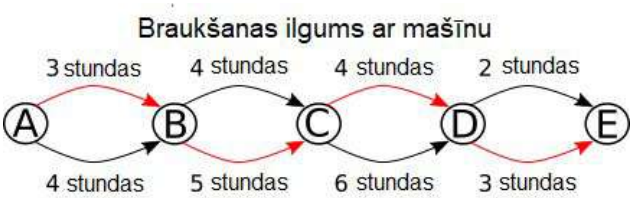
Pimais: 9 stundas ar mašīnu



Braukšanas ilgums ar vilcienu

No/uz	Mašīna	Vilciens	Stundas
A uz B	Mašīna 3 stundas		3
B uz C	Mašīna 4 stundas		4
C uz D		Vilciens 6 stundas	6
D uz E	Mašīna 2 stundas		2
Pavadītais laiks	9 Stundas	6 Stundas	15

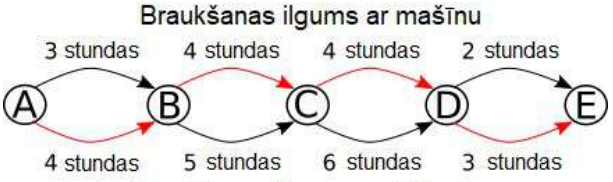
Otrais: 7 stundas ar mašīnu



Braukšanas ilgums ar vilcienu

No/uz	Mašīna	Vilciens	Stundas
A uz B	Mašīna 3 stundas		3
B uz C		Vilciens 5 stundas	5
C uz D	Mašīna 4 stundas		4
D uz E		Vilciens 3 stundas	3
Pavadītais laiks	7 Stundas	8 Stundas	15

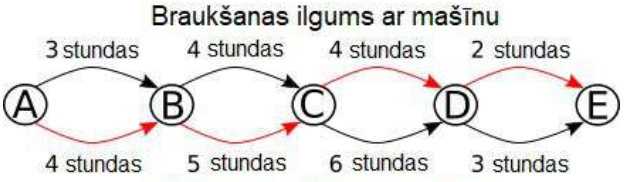
Trešais: 8 stundas ar mašīnu



Braukšanas ilgums ar vilcienu

No/uz	Mašīna	Vilciens	Stundas
A uz B		Vilciens 4 stundas	4
B uz C	Mašīna 4 stundas		4
C uz D	Mašīna 4 stundas		4
D uz E		Vilciens 3 stundas	3
Pavadītais laiks	8 Stundas	7 Stundas	15

Ceturtais: 6 stundas ar mašīnu



Braukšanas ilgums ar vilcienu

No/uz	Mašīna	Vilciens	Stundas
A uz B		Vilciens 4 stundas	4
B uz C		Vilciens 5 stundas	5
C uz D	Mašīna 4 stundas		4
D uz E	Mašīna 2 stundas		2
Pavadītais laiks	6 Stundas	9 Stundas	15

No A uz E ir iespējams nokļūt arī ātrāk, bet tādā gadījumā Jānim ir jāpavada vairāk laika braucot ar mašīnu.

Kā tas ir saistīts ar informātiku?

Šāda veida uzdevumi tiek saukti par optimizācijas uzdevumiem. Ir dots noteikts ierobežojumu kopums (šajā gadījumā - ceļojumam kopējais atvēlētais laiks) un ir kāds lielums, kuru pie šiem ierobežojumiem nepieciešams maksimizēt vai minimizēt (šoreiz nepieciešams minimizēt mašīnas izmantošanas laiku).

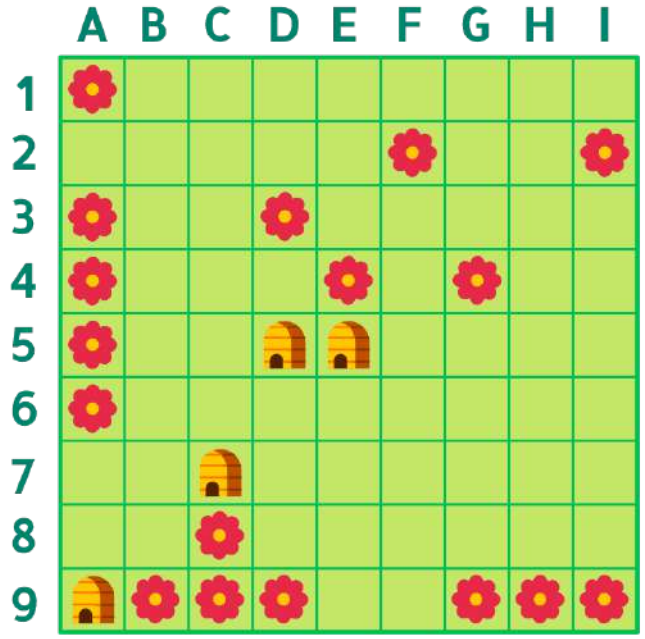
Lai atrisinātu šāda veida uzdevumus, bieži ir izdevīgi to pārfrāzēt vieglāk atrisināmā uzdevumā - kā tas ir darīts arī šoreiz.

Dotais uzdevuma risinājums ir saistīts arī ar datordomāšanu. Uzdevumā dotie dati tiek strukturēti tabulu veidā, kur pirmajā tiek aprēķināts katrā posmā ietaupītais laiks, ceļojot ar mašīnu vilciena vietā. Pēc tam šī informācija tiek izmantota, lai iegūtu četrus derīgus risinājumus un atrastu optimālo (ar mazāko mašīnas izmantošanas laiku).

Bišu strops

Pakistāna

Biškopim ir strops ar bitēm. Viņš vēlas stropu novietot tā, lai attāluma summa starp stropu un līdz katrai no puķēm būtu pēc iespējas mazāka. Lauks ar ziediem ir attēlots zemāk redzamajā rūtiņu tabulā, kuras rindas ir sanumurētas no 1 līdz 9 un kolonnas apzīmētas ar burtiem no A līdz I.



Bites šajā laukā lido tikai horizontāli un vertikāli, tāpēc attālums starp divām rūtīm ir horizontālā un vertikālā attāluma summa. Piemēram, attālums starp C4 un D7 ir 4 (3 rūtis vertikāli un 1 rūts horizontāli).

Uzdevums:

Kur biškopim būtu jānovieto strops, lai attālumu summa no stropa līdz katrai puķei ir pēc iespējas mazāka? (iespējamās vietas kartē ir iezīmētas ar)

Atbilžu varianti:

A) D5
B) C7
C) E5
D) A9

Pareizā atbilde: A) D5

Problēma var tik atrisināta, atrodot puķu skaita virkņu mediānas horizontāli un vertikāli. Tā kā kopā šajā laukā ir 17 puķes, vidējā puķe sanāk devītā pēc kārtas (atstājot 8 puķes

vienā un 8 puķes otrā pusē). Sakārtojot vertikāli, mediāna ir puķe rūtī A5, tātad strops būtu jāievieto 5.rindā. Sakārtojot horizontāli, mediāna sanāk vai nu puķe rūtī D3, vai D9, jo tās abas ir vienā kolonnā. Tātad strops jāieliek D kolonnā. Tas mūs noved līdz optimālajai stropa novietošanas vietai, kas ir D5.

Tā kā par attālumu ir uzskatīta vertikālā un horizontālā attāluma summa (saukta arī par Manhetenas attālumu, nevis taisnā līnijā (Eiklīda attālums)), problēma var tikt atrisināta, atsevišķi atrodot stropa novietojuma kolonnu un atsevišķi - rindu, jo novietojums vienā virzienā neietekmē kopējo attālumu otrāi. Turklāt algoritms, kas tiek izmantots optimālās kolonnas atrašanai, var tikt izmantot arī rindas atrašanai, un tādā veidā optimālā stropa atrašanās vieta būs tā, kur abas šīs koordinātas tiks sakombinētas.

Kā tas ir saistīts ar informātiku?

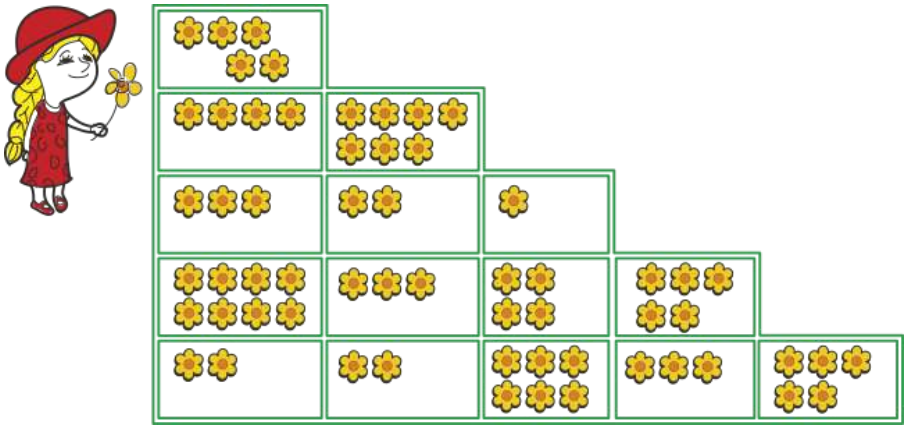
Šī uzdevuma paredzētais risināšanas veids (nerēķinot visus iespējamo ceļu garumus) izmanto “skaldi un valdi” pieeju, jo sākotnējais uzdevums tiek sadalīts divos apakšuzdevumos: atrast optimālo rindu un atrast optimālo kolonu. Pēc tam, kad abi apakšuzdevumi ir atrisināti, iegūtie rezultāti tiek apvienoti vienā kopējā rezultātā.

https://en.wikipedia.org/wiki/Divide-and-conquer_algorithm

Sarkangalvīte

Rumānija

Sarkangalvīte vēlas salasīt puķes no vecmāmiņas dārza. Dārzs ir sadalīts vairākās daļās, kur katra daļa satur konkrētu puķu skaitu. Sarkangalvīte sāk savu ceļu no augšējā kreisā stūra un tad dodas lejup uz apakšējo labo stūri, ejot tikai uz leju vai pa labi.



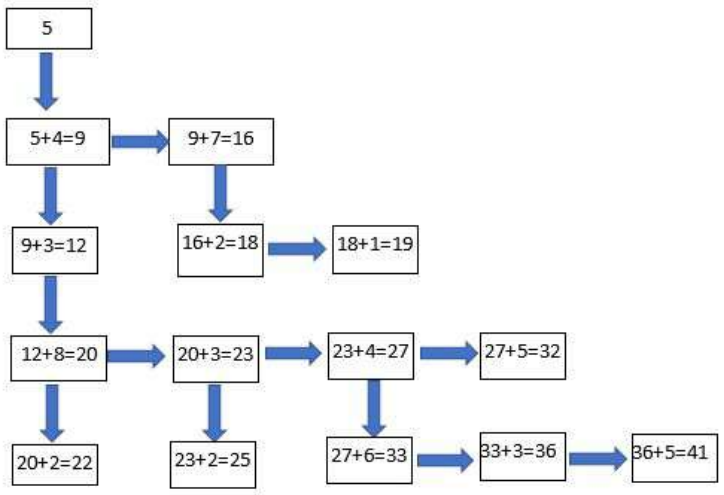
Uzdevums:

Kāds ir lielākais ziedu skaits, kurus viņa var salasīt sava gājiena laikā?

Pareizā atbilde: 41

Katrā daļā mēs varam izrēķināt maksimālo puķu skaitu, kuru Sarkangalvīte var iegūt no sākotnējās ailes līdz tai daļai (sauksim to par maksimumu). Maksimumu var izrēķināt šādi:

- Ja daļu var sasniegt tikai no vienas iepriekšējās daļas, tad tās maksimums ir ziedu skaita summa šajā daļā un iepriekšējās daļas maksimums.
- Ja daļu var sasniegt no divām iepriekšējām daļām, tad tās maksimums ir puķu skaits tajā daļā, kam pieskaitīts lielākais maksimums no divām iepriekšējām daļām.



Kā tas ir saistīts ar informātiku?

Šis ir dinamiskās programmēšanas uzdevums. Šādos uzdevumos labākais atrisinājums tiek iegūts, “pa ceļam” atrisinot šī uzdevuma apakšuzdevumus un šos risinājumus izmantojot gala rezultāta iegūšanai.

Šajā uzdevumā starprezultāti ir lielākais puķu skaits, ko Sarkangalvīte var savākt līdz katrai dārza daļai virzoties no sākuma līdz beigu daļai. Gudrība slēpjas tajā, ka nav nepieciešams šo skaitli rēķināt no jauna, bet var izmantot iepriekš iegūto rezultātu un lielāko līdz konkrētajai daļai savākto puķu skaitu var iegūt kā lielāko no divām summām, katrā solī veicot vienu salīdzināšanu un vienu saskaitīšanu. Ar šī algoritma palīdzību brīdī, kad nonāksim pēdējā daļā, mums jau būs aprēķināta vajadzīgā atbilde.

Daudzi praktiski optimizācijas uzdevumi tiek risināti ar šādu, laika ziņā efektīvu, risināšanas algoritmu.

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

Noliktavas

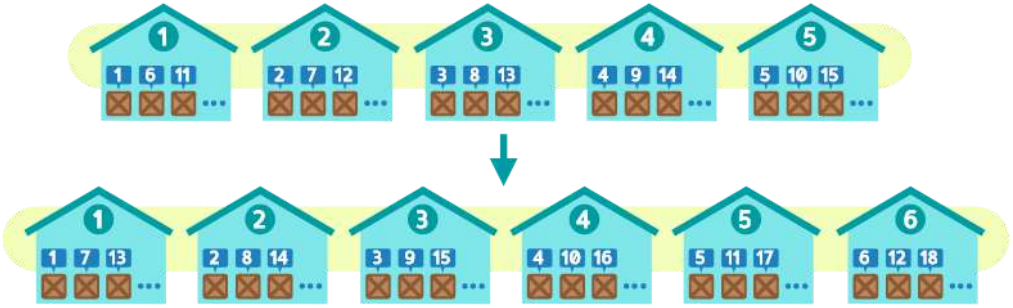
Krievija

42 eži novieto savas mantas 5 noliktavās. Pirmais ezis novieto savas mantas pirmajā noliktavā, otrais otrajā, ..., sestais atkal pirmajā noliktavā un tā tālāk.

Vienu dienu eži uzbūvēja jaunu noliktavu, un tā tika atzīmēta ar ciparu 6. Viņi nolēma pārvietot savas mantas noliktavās tā, lai sadalījums joprojām būtu vienkāršs. Pirmais ezis novieto savas mantas pirmajā noliktavā, otrais otrajā, ..., septītais atkal pirmajā noliktavā un tā tālāk.

Uzdevums:

Cik ežiem nebija jāpārvieto mantas jaunajā veidā?



Pareizā atbilde: 10

Eži numur 1, 2, 3, 4, 5 un 31, 32, 33, 34, 35 savas mantas nepārvietoja. Lai to noskaidrotu mums, pirmkārt, ir nepieciešams vienkāršot numerāciju: jānumurē eži un noliktavas no 0. Tādā veidā mēs iegūstam ežus no 0 līdz 41. Un ezim numur x nevajadzētu pārvest savas mantas, ja x mod 5 = x mod 6. Tas ir iespējams tad, ja 30 | x - r, kur r = x mod 5 = x mod 6. Tātad, tas ir iespējams tikai ar x = 0 + 0, 1, 2, 3, 4, un x = 30 + 0, 1, 2, 3, 4.

Kā tas ir saistīts ar informātiku?

Šajā uzdevumā notiek sadalīta resursu glabāšana. Lai noteiktu, kur katrs resurss jānovieto vai jau atrodas, iespējams lietot jaucējadresēšanu jeb jaukšanu. Katram resursam tiek aprēķināta jaucējatslēga, kas norāda vietu, kur attiecīgais resurss jānovieto vai ir atrodams. Piemēram, ja resursi ir cilvēki, kuru vārdi mums ir zināmi, tad kā jaucējatslēgu var izmantot vārda pirmo burtu. Ja rodas vajadzība atrast Matīsu, tad vispirms atrodam, kādi cilvēki ar "M" mums ir zināmi (piemēram, vēl arī Matlide, Mareks un Mikus) un tad informāciju par Matīsu meklējam jau šajā ierobežotajā (visiem viena jaucējatslēga) kopā.

Uzdevumā jaucējatslēga tiek aprēķināta kā atlikums, dalot ar veselu skaitli, un atslēgas vērtība tiek mainīta līdz ar jaunās noliktavas uzcelšanu. Labas jaucējfunkcijas, kuras datu tabulas lieluma (jauna noliktava šajā uzdevumā) izmaiņu dēļ izsauc nepieciešamību pārvietot tikai ierobežotu datu apjomu, sauc par konsistentām jaucējfunkcijām.

Skaitītājs

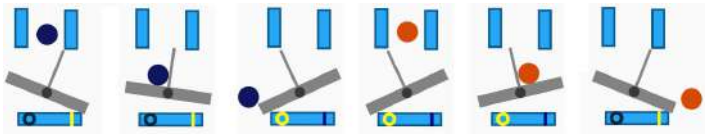
Austrija

Iekārtā ir 4 dēlīši, kas, līdzīgi kā šūpolēs, var būs noliekušies pa labi vai pa kreisi

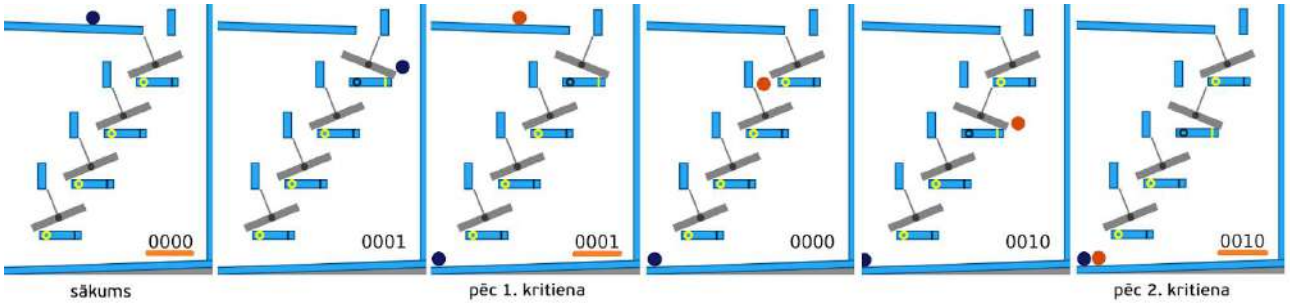
Dēlītis, kas noliecies pa kreisi = 0

Dēlītis, kas noliecies pa labi = 1

Kad krītoša bumbiņa nonāk uz kāda no dēlīšiem, tas noliecas un bumbiņa noripo.



Zemāk redzama iekārtas darbība, kad tiek ripinātas 2 bumbiņas.
Pirmajā attēlā visi dēlīši ir pa kreisi, tāpēc skaitītājs rāda 0000:



Uzdevums:

Kāds būs skaitītāja rādījums pēc 5 nomestām bumbiņām?

Pareizā atbilde: 0101

Skaitītāja rādījumi mainās šādi: 0001, 0010, 0011, 0100, 0101
Šī ierīce ir binārais skaitītājs.

Sākuma pozīcija ir 0000.

Pirmā bumbiņa noliec 1.dēlīti pa labi. Skaitītājs rāda 0001.

Otrā bumbiņa noliec 1.dēlīti pa kreisi, bet 2. - pa labi. Skaitītājs rāda 0010.

Trešā bumbiņa noliec 1.dēlīti pa labi un noripo. 2.dēlītis paliek noliecies pa labi. Skaitītājs rāda 0011.

Ceturtā bumbiņa 1.dēlīti noliec pa kreisi, 2.dēlīti arī pa kreisi, bet 3.dēlīti pa labi. Skaitītājs rāda 0100.

Pēdējā bumba 1.dēlīti noliec pa labi un noripo nost. 2.dēlītis paliek noliecies pa kreisi, bet 3. - noliecies pa labi. Skaitītājs rāda 0101.

Atbilde jāsniedz kā četru ciparu (katrs 0 vai 1) virkne.

Kā tas ir saistīts ar informātiku?

Ja skaitļu pierakstam tiek izmantoti tikai divi cipari (parasti - 0 un 1), tas nozīmē, ka tiek izmantota divnieku jeb binārā skaitīšanas sistēma, kas jau no pirmsākumiem ir ļoti populāra datoros, jo viegli iedomāties un tehniski realizēt situāciju, kad katrs elements ir tikai vienā no diviem stāvokļiem - ieslēgts (1) vai izslēgts (0).

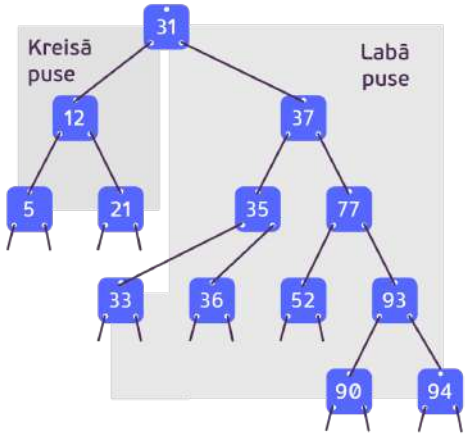
Šajā uzdevumā aprakstītā mašīna apraksta mehānisku bināru četrciparu skaitītāju, kas var skaitīt līdz 15. Visi četrciparu binārie skaitļi ir uzskaitīti tabulā:

Binary number	Number
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

Varat noskatities video, kā šim līdzīgs binārs skaitītājs darbojas: https://www.youtube.com/watch?v=zELAfmp3fXY&fbclid=IwAR1iVWNrmrj-bNzD26i5xNivAaN2Ue7o_AWkRtIpRGamiHFEsOfigpRWizQ

Numurētās kārtis

Holande



No numurētām kārtīm ir izveidota struktūra. Katra kārts ir veidota šādi:

Tai augšpusē ir caurums, bet apakšā divas virves, pie kurām var piestirpināt citas kārtis, dēvētas par pēcteča kārtīm.

Uz katras kārts ir uzrakstīts vesels skaitlis N.

Ja numurētā kārts ir piestiprināta pie kreisās virves, tās un tās pēcteču kāršu numuriem ir jābūt mazākiem par skaitli N.

Ja numurētā kārts ir piestiprināta pie labās virves, tās un tās pēcteču kāršu numuriem ir jābūt lielākiem par skaitli N.

Uzdevums:

Šobrīd struktūrā kārtīm kopā ir 14 brīvas virves. Pie cik no šīm virvēm lielākais vēl var pievienot kārtis, lai pagarinātu doto struktūru?

Pareizā atbilde: 11

Ir 14 tukšas virves. Kārtij ar numuru 36 vairs nevar pievienot nevienu citu kārti. Kreisās pēcteča kārts numuram būtu jābūt lielākam par 35 un mazākam par 36, bet šāds vesels skaitlis neeksistē. Labās puses pēcteča kārtij būtu jābūt lielākai par 36 un mazākai par 37, bet arī šāds vesels skaitlis neeksistē. Kārtij ar numuru 94 var pievienot tikai labo pēcteča kārti.

Kā tas ir saistīts ar informātiku?

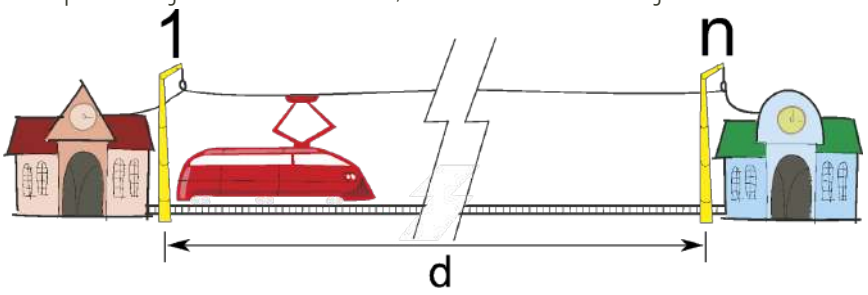
Uzdevumā aprakstītā struktūra ir pazīstama kā binārais meklēšanas koks. Tas izskatās kā otrādi apgriezts koks. Par bināru to sauc tāpēc, ka katrai kārtij ir piestiprinātas tieši divas virves (priedēklis “bi-” nāk no latīņu valodas ar nozīmi “divi”) un, tātad, katrai kārtij var piesaistīt ne vairāk kā divas citas kārtis. Šādās datu struktūrās var glabāt visdažādākā tipa datus un to priekšrocība ir iespēja ātri (logaritmiskā laikā) atrast tajā esošos datus.

Dzelzceļa robots

Čehija

Dzelzceļa līnija starp divām stacijām ir apgādāta ar elektrību. Mastiem, kas piegādā elektrību, ir jābūt novietotiem regulāros intervālos - pirmajam mastam jāatrodas pirmajā stacijā, bet pēdējam - otrajā.

Dzelzceļa robots darbojas pēc programmas, kas redzama zemāk. Attālums starp stacijām ir **d** metri, un robotam ir jānovieto **n** masti.



Uzdevums:

Pievieno derīgos parametrus programmai tā, lai robots varētu strādāt pareizi!

Aizvelc pareizo izteiksmi uz pareizo kastīti programmā pa labi!

Atbildes:

n	n+d
d	n−d
1	d−n
2	d*n
n+1	n/d
d+1	d/n
n−1	n/(d+1)
d−1	d/(n+1)
n/2	n/(d−1)
d/2	d/(n−1)

Pareizā atbilde: n, n-1, d/(n-1)

Ja robotam ir jānovieto n masti, viņam sākumā ir arī jāuzlādē n masti. Pēc tam, kad viens masts ir novietots, n-1 masts ir palicis pāri. Tātad robotam ir jāatkārto n-1 reizes pārvietošanās un viena masta novietošana. Tad mums ir jāsadala attālums d ar (n-1), lai

saprastu, cik daudz robotam ir nepieciešams pārvietoties, lai novietotu nākamo mastu, tādējādi (n-1) * [d/(n-1)] = d.

Kā tas ir saistīts ar informātiku?

Uzdevumā dotais koda fragments ir procedūra, kurā tiek izmantoti mainīgie d un n. Mainīgā vārds apzīmē kādu skaitlisku vērtību, kas laika gaitā var mainīties. Mainīgie d un n ir definēti iepriekš galvenajā programmā un tiek izmantoti procedūrā. Šādos gadījumos mainīgos sauc par globāliem mainīgajiem un tiem var piekļūt no atšķirīgām procedūrām šajā programmā.

Uzdevumā tiek izmantoti arī cikli - instrukciju virknes (cikla ķermenis), kas tiek atkārtotas vairākkārt. Šajā uzdevumā ir svarīgi noskaidrot, cik reizes cikla ķermenis jāizpilda. Šoreiz, lai uzstādītu visus mastus, cikla ķermenis jāizpilda n-1 reizi.

Procedūras un funkcijas ir būtisks programmēšanas valodu koncepts, tās tiek izmantotas specifisku apakšuzdevumu veikšanai un to darbības raksturu var vadīt ar tām nododamo parametru vērtībām.

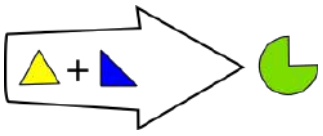
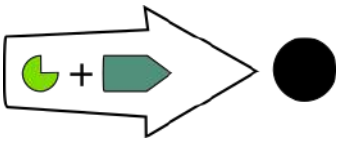
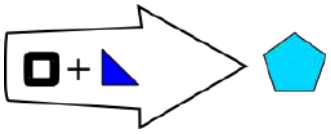
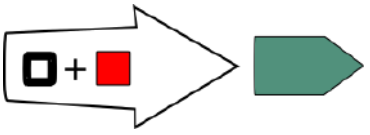
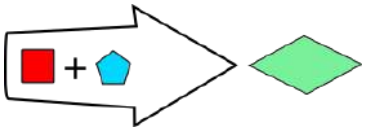
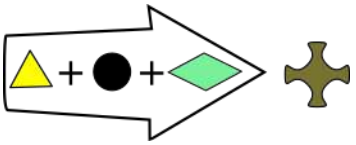
Ķimikālijas

Ungārija

Trīs zinātnieki darbojas ar ķimikālijām un viņu mērķis ir no četrām ķimikālijām izveidot jaunu ķimikāliju .

Katrs zinātnieks strādā divas dienas. Ik dienas strādā tikai viens zinātnieks, un viņš vai viņa var izpildīt tikai vienu darbību dienā.

Viena darbība nozīmē savienot divas ķimikālijas, izveidojot jaunu, kuru pēc tam var izmantot jebkuram skaitam darbību nākotnē. Katrs zinātnieks var veikt tikai divu veidu darbības:

Ķīmiķe Karlija (C)		
Fiziķe Paula (P)		
Biologs Berijs (B)		

Uzdevums:

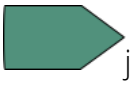
Sekojošā burtu secība norāda uz secību, kādā zinātnieki strādā. Kura no darbību secībām neļautu zinātniekiem izveidot ķimikāliju ?

Atbilžu varianti:

A) C P B P C B
B) P P C C B B
C) C P B C P B
D) P B P C C B

Pareizā atbilde: C

Tā kā B (biologa Berija) otrajā darbībā izveidojas gala produkts, viņam jābūt pēdējam sarakstā. Lai izveidotu gala produktu, izmantojot savu otro darbību, viņam būtu jābūt  .

Lai izveidotu melno  apli , C (ķīmiķei Karlijai) ir  jābūt pieejai un ir jādodas uz laboratoriju pēc fiziķes Paulas. Tāpat arī C ir jāveic sava pirmā darbība pirms otrās darbības. Lai veiktu savu pirmo darbību, B ir nepieciešams, lai pirms viņa P izpilda savu pirmo darbību.

Atbildē "C" pirmais P secībā var izveidot tikai vienu no nepieciešamajām ķimikālijām, kas nepieciešamas nākamo darbību veikšanai. Vienai no tām ir jāizgāžas. Secība C P B C P B veiksmīgi nevarēs izveidot nepieciešamo produktu.

Kā tas ir saistīts ar informātiku?

Datorzinātnē un programmēšanā tiek veidoti un izmantoti procesi, kuriem ir noteikta veida ievaddati un pēc kuru izpildes tiek iegūti noteikta veida izvaddati. Ja viens process ir atkarīgs no cita procesa rezultāta, tad šo procesu nedrīkst sākt, ja tam nepieciešamie dati vēl nav sagatavoti. Programmētājam ir jānodrošina tāda procesu vadība, kuras laikā visi procesi tiek uzsākti tikai tad, kad iepriekšējie procesi ir beigušies un visi nepieciešamie dati ir sagatavoti.

Robota zīmējums

Slovākija

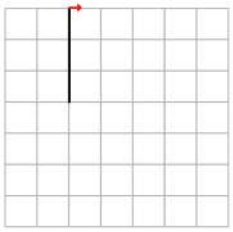
Robots, kustoties pa kvadrātveida režģi, ar līnijām veido attēlu. Katru attēlu apraksta trīs ciparu virkne.

Piemēram, "3,1,5" apraksta 4.zīmējumā redzamo figūru, jo nozīmē:

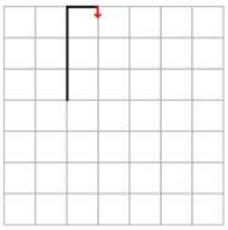
Pārvietoties 3 vienības (rūtiņas malas garums) uz priekšu, tad pagriezties pa labi (1.zīm.)

Pārvietoties 1 vienību uz priekšu, tad pagriezties pa labi (2.zīm.)

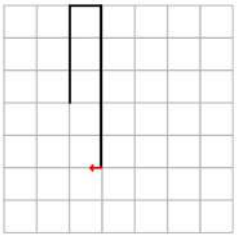
Pārvietoties 5 vienības uz priekšu, tad pagriezties pa labi (3.zīm.).



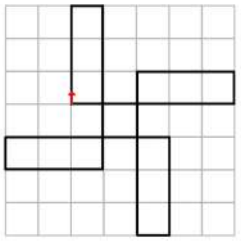
1.zīmējums



2.zīmējums



3.zīmējums



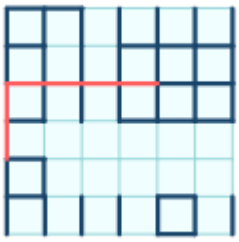
4.zīmējums

Dotā darbību virkne tiek atkārtota bezgalīgi.

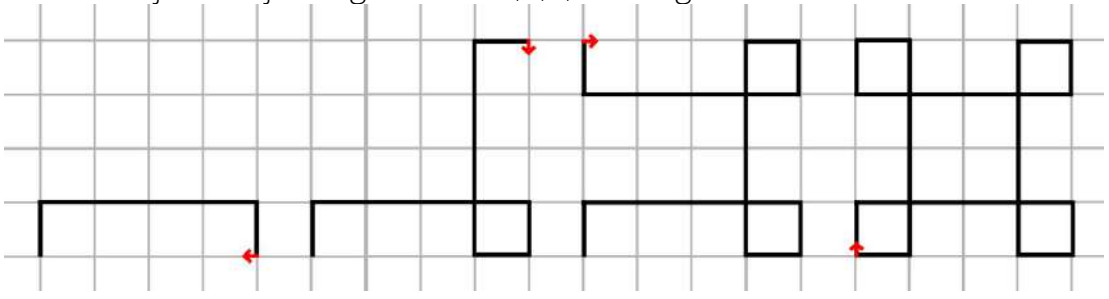
Uzdevums:

Robots ir sācis zīmēt figūru, kuru apraksta "2,4,3". Klikšķini uz līnijām, lai uzzīmētu šo figūru pilnībā!

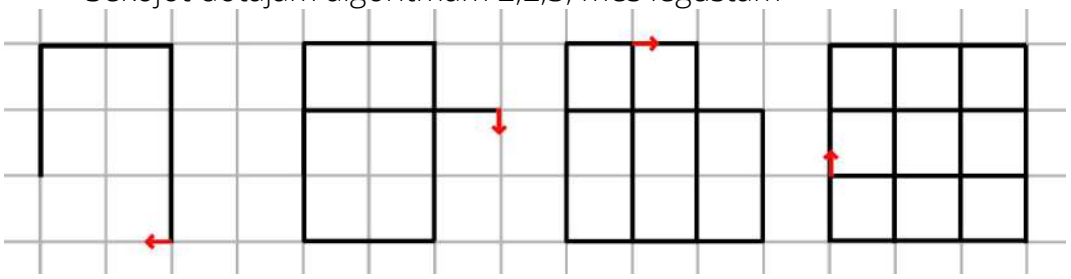
Pareizā atbilde:



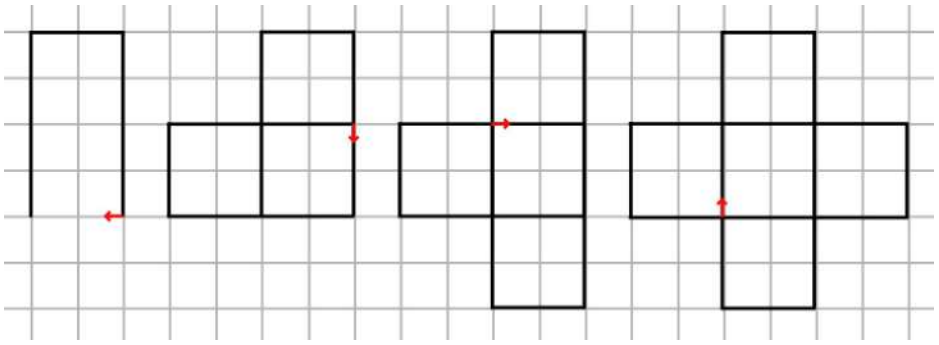
Sekojot dotajam algoritmam 1,4,1, mēs iegūstam



Sekojot dotajam algoritmam 2,2,3, mēs iegūstam



Sekojot dotajam algoritmam 4,2,4, mēs iegūstam



Sekojot dotajam algoritmam 3,3,3, mēs iegūstam 3x3 izmēra kvadrātu.

Kā tas ir saistīts ar informātiku?

Datorprogrammu izpildes gaitā datorā tiek izpildītas datorprogrammas instrukcijas. Katra instrukcija satur noteiktu veicamās darbības aprakstu. Darbības izpilde izraisa noteiktu efektu, kas saistīts ar izpildītās darbības semantiku. (https://en.wikipedia.org/wiki/Execution_(computing))

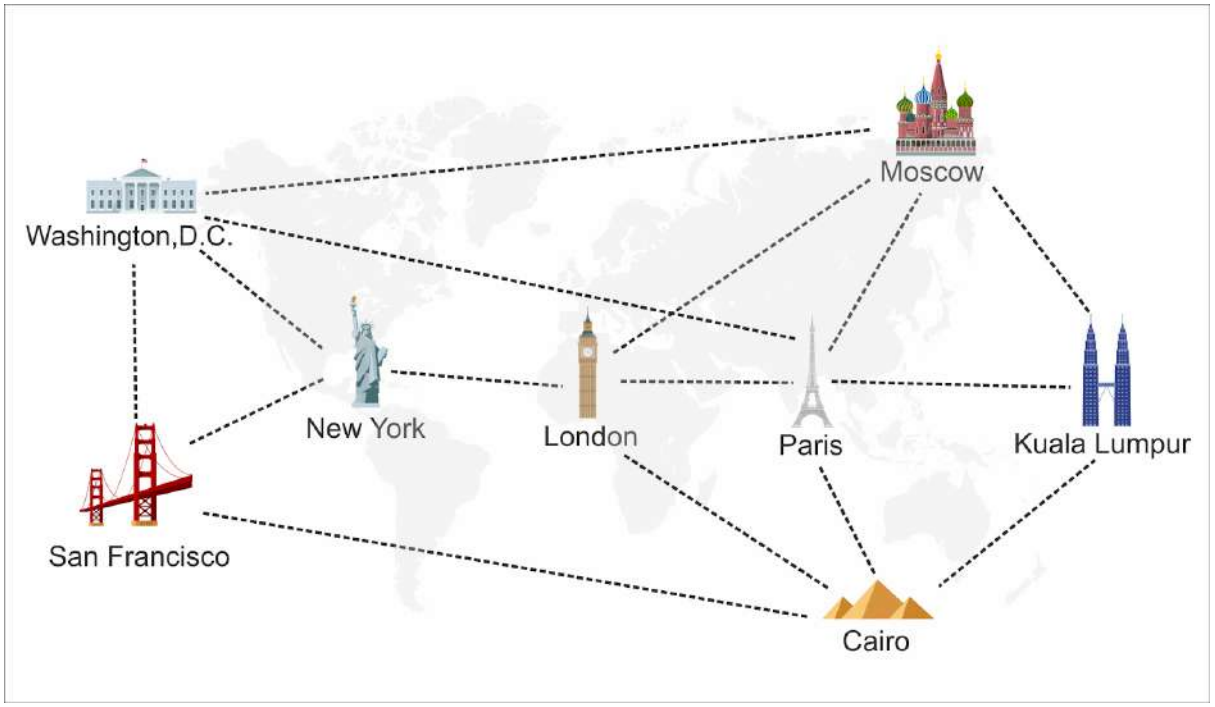
Katra uzdevuma atrisināšana ietver instrukciju semantikas un aprakstītā algoritma izpildes gaitas izpratni, kas ir neatņemama kvalitatīvas programmēšanas sastāvdaļa.

Jums jāspēj izlasīt un saprast instrukciju aprakstu, kā arī izpildīt tās soli pa solim, lai pārliecinātos par izraisītajiem efektiem. Šī ir ļoti būtiska laba programmētāja spēja, kas diendienā tiek izmantota datorprogrammu atkļūdošanas procesā.

Videi draudzīgāki lidojumi

Beļģija

Bebras Starptautiskajai aviosabiedrībai ir daudz lidojuma maršrutu, kas savieno vairākas lielās pilsētas, kā tas ir parādīts attēlā:



CO_2 piesārņojums ir galvenais globālās sasilšanas cēlonis. Lai mazinātu šo gaisa piesārņojumu, aviosabiedrība vēlas likvidēt dažus no mašrutiem tā, lai klienti joprojām varētu nonākt jebkurā no pilsētām.

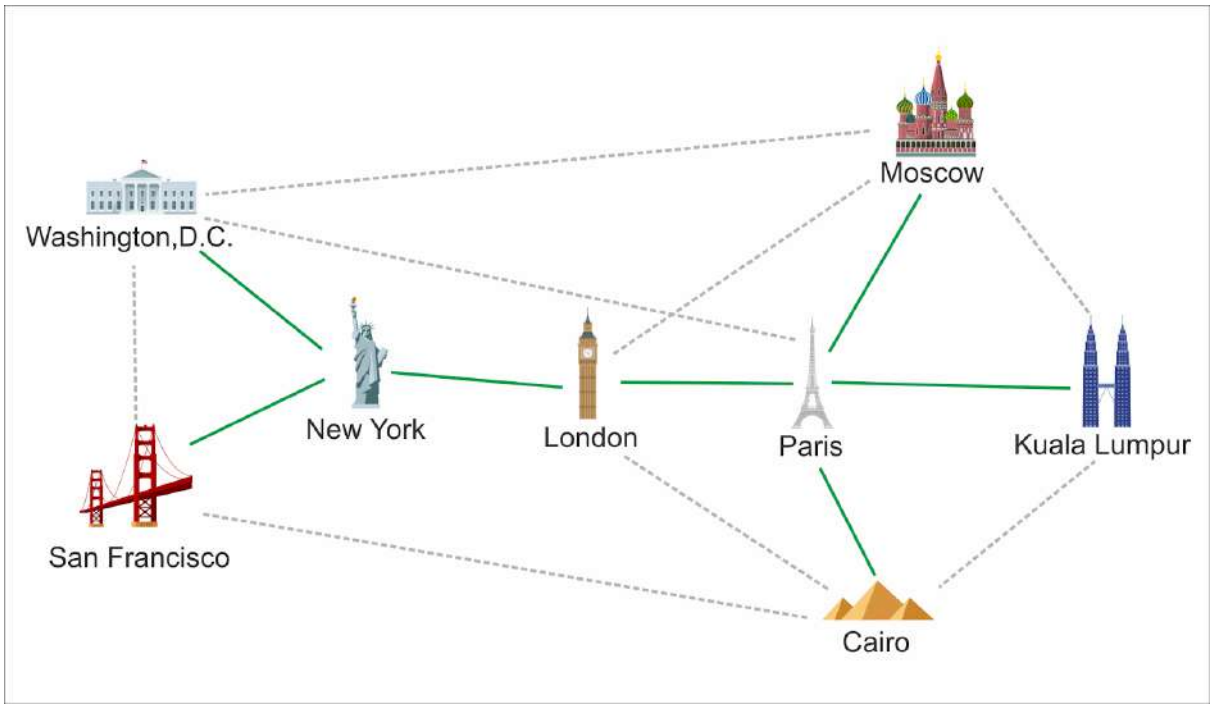
Piemēram, ja likvidētu lidojumu starp Sanfrancisko un Vašingtonu, klienti varētu lidot no Sanfrancisko uz Ņujorku, un tad no Ņujorkas uz Vašingtonu.

Uzdevums:

Kāds ir lielākais maršrutu skaits no augstāk parādītajiem lidojuma ceļiem, kurus aviosabiedrība var likvidēt?

Pareizā atbilde ir: 8

Zemāk redzamais piemērs parāda, ka ir iespējams atcelt 8 maršruta līnijas, joprojām ļaujot klientiem nokļūt uz jebkuru pilsētu



Ievēro, kā, likvidējot 8 maršrutus, mums joprojām paliek 15-8=7 maršruti. Kāpēc mēs nevaram likvidēt vairāk maršrutu? Ja būtu tikai 2 pilsētas, būtu skaidrs, ka mums ir nepieciešams 1 maršruts. Ja būtu 3 pilsētas, tad nepieciešami būtu 2 maršruti. Tā šis modelis turpinās, un 8 pilsētām ir nepieciešami 7 maršruti.

Lai pārliecinātos, ka šis modelis ir pareizs, mums ir jāsaprot, ka gadījumā, ja likvidējam 9 vai vairāk maršrutus, klients ne vienmēr varēs nokļūt uz jebkuru pilsētu. Ja 9 vai vairāk maršruti ir likvidēti, tad paliks vairs tikai 6 vai mazāk maršruti. Kas notiek gadījumā, ja ir tikai 6 maršruti? Tā kā kopā ir 8 pilsētas, 6 maršrutu gadījumā kāda pilsēta paliek vai nu vispār bez maršruta līnijas, vai nu ar tieši vienu maršruta līniju. Ja ir pilsēta ar tieši vienu maršruta līniju, tad mēs varam noņemt šo pilsētu un maršrutu, paliekot ar 5 maršrutiem un 7 pilsētām. Turpinot šo, var secināt, ka visos gadījumos mēs paliksim ar kādu pilsētu bez maršruta vai galu galā bez maršruta starp 2 pilsētām. Tāpēc, ja ir palikuši vien 6 maršruti, klients ne vienmēr varēs aizlidot uz jebkuru pilsētu.

Kā tas ir saistīts ar informātiku?

Lidmašīnu maršrutus ir dabīgi attēlot grafa veidā. Šādā grafā virsotnes atbilst pilsētām, bet šķautnes - lidmašīnu maršrutiem. Uzdevums grafu terminos nozīmē atrast minimālo pārklājošo koku - no sākotnējā grafa izmest lielāko iespējamo šķautņu skaitu, lai atlikušais grafs joprojām būtu sakarīgs.

Vairāk informācijas: https://en.wikipedia.org/wiki/Minimum_spanning_tree.

Minimālā pārklājošā koka meklēšana ir tradicionāls uzdevums dažādu tīklu (telekomunikāciju, transporta, ūdensvadu, ...) izpētes un veidošanas procesā.

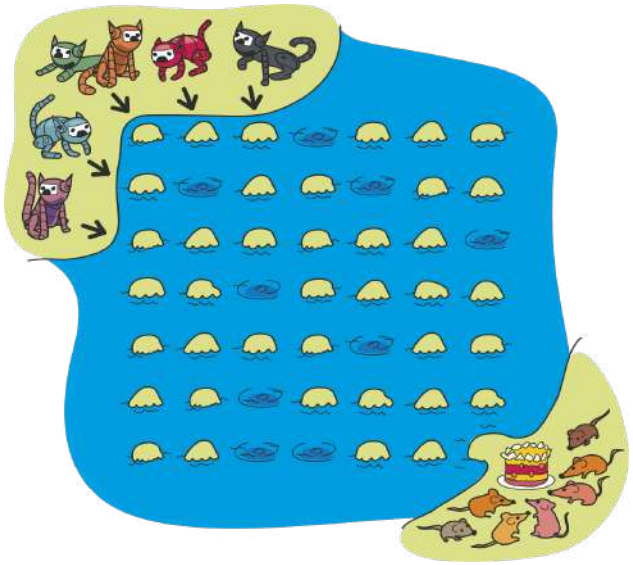
Peļu ballīte

Lietuva

Seši kaķi dodas uz peļu ballīti, lecot no salas uz salu. Kaķi var lēkt tikai uz blakus esošo salu. Viņi nevar lēkt pa diagonāli, kā arī uz vai pāri salām, kas ir zem ūdens.

Kaķis lec ļoti ātri, tāpēc varam uzskatīt, ka laiks, kas nepieciešams lēcienam, ir nulle. Pēc lēciena kaķis ir ļoti noguris, un tam ir nepieciešama atpūta, tāpēc kaķis paliek uz salas vienu minūti un tad atkal lec uz nākamo salu, ja tā nav jau aizņemta. Citādi kaķis paliek uz tās pašas salas vēl vienu minūti.

Kaķis var uzlēkt uz salas tikai tad, ja iepriekšējais kaķis dodas tālāk. Kaķi var sākt no jebkuras ar bultiņu norādītās salas.

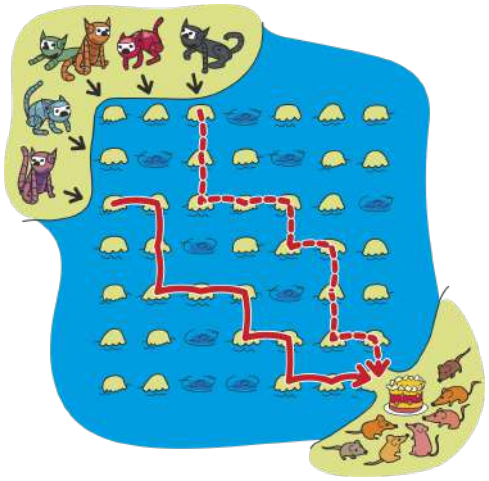


Uzdevums:

Cik minūtēs visi kaķi varēs nokļūt no sākumpunkta līdz galamērķim (ātrākajā variantā)?

Pareizā atbilde ir: 12

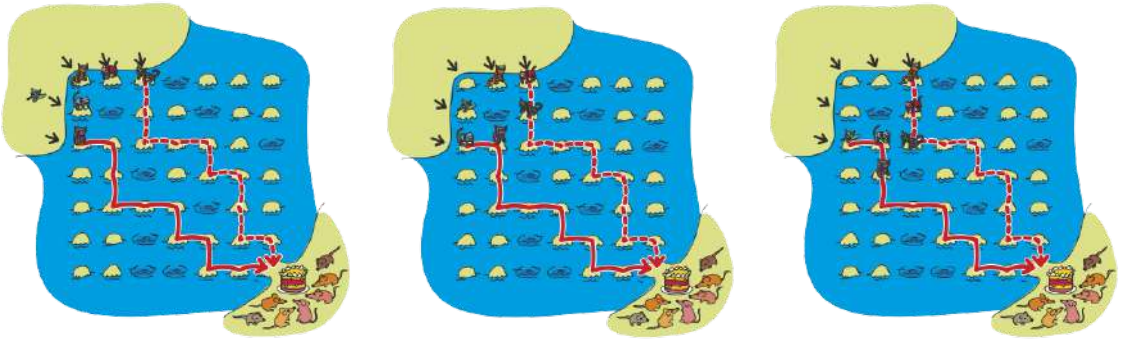
No sākuma mums ir jāatrod īsākie ceļi no visiem iespējamajiem ieejas punktiem - tie ir 10 minūtes, 11 minūtes un 12 minūtes gari. Tad vēl svarīgāk ir piezīmēt, ka ir tieši divi dažādi īsākie ceļi (lai dotos divās paralēlās plūsmās), kā tas ir parādīts attēlā:



Tātad 5 kaķi var pagūt nonākt galamērķī 12 minūtēs. Pēdējais sestais kaķis var sākt doties vienlaicīgi ar piekto kaķi, izmantojot vienu no salām tuvāk stūrim.

Došanās uz ballītes salu kaķim aizņem 10 minūtes. Ja ir divi kaķi, laiks joprojām ir 10 minūtes. Trīs kaķi nevar vienlaicīgi izmantot īsākos paralēlos ceļus, jo tie ir tikai divi. Tāpēc trešajam kaķim ir jāuzsāk savs ceļš vai nu pēc pirmā vai otrā kaķa cauruļvada manierē, un tas aizņems 11 minūtes. Līdzīgi tam, arī četriem kaķiem tas aizņems tikpat daudz laika kā trim kaķiem. Loģiski spriežot, pieciem vai sešiem kaķiem tas aizņems 12 minūtes šajā pašā gadījumā.

Pirmais solis Otrais solis Trešais solis



Padziļinātāks skaidrojums:

Kāpēc viņi to nevar izdarīt ātrāk?

Katram kaķim būtu jāsāk ceļš no pirmās vai pēdējās ieejas salas. Pēc tam, ja kaķis sāk no kādas citas salas un šķērso tālāk kādu no šīm divām, mēs varam uzskatīt, ka kaķis savu ceļu sācis no šejienes, aizmirstot iepriekšējos ceļa posmus. Tāpēc tagad mēs esam izveidojuši šo uzdevumu par uzdevumu ar tikai divām salām.

Katrai no šīm divām salām ir acīmredzams īsākais desmit minūšu ceļš uz peļu ballīti, kā redzams attēlā. Tāpēc mēs varam sadalīt kaķus divās plūsmās. Tā kā būtu jābūt vismaz trim kaķiem katrā plūsmā, mums ir jāpievieno vismaz vēl 2 minūtes otrajam un trešajam kaķim katrā plūsmā, tātad mums vajag vismaz 12 minūtes.

Ja trīs kaķi sāk no vienas no mūsu divām sākuma salām viens pēc otra, un trīs pārējie kaķi sāk no otras sākuma salas tajā pašā laikā, būs nepieciešamas tieši 12 minūtes.

Kā tas ir saistīts ar informātiku?

Datorzinātniekiem jāmāk modelēt un novērtēt dažādas situācijas, kas bieži tiek pierakstītas grafu terminos. Viens no uzdevumiem ir izprast un paredzēt matemātiski modelētas sistēmas uzvedību, kā arī paredzēt cik efektīvi būs realizētie algoritmi un novērtēt to ātrdarbību. Optimizācijas uzdevumos nepieciešams atrast vislabāko starp daudziem pieņemamiem.

Datorzinātnē ir būtiski spēt "pārtulkot" reālās dzīves uzdevumus datoram piemērotās un izmantojamās datu struktūrās. Šoreiz dati doti kā režģis, kurā var pārvietoties tikai vertikālā vai horizontālā virzienā, izvairoties no salām, kas atrodas zem ūdens.

Šo uzdevumu var uzskatīt par grafu teorijas uzdevumu, kurā salas (ieskaitot ieejas un ballītes vietu) tiek modelētas kā grafa virsotnes, kur ar šķautnēm ir saistītas tās virsotnes, kurām atbilstošās salas atrodas blakus. Šādā pieejā nogrimušās salas vienkārši tiek ignorētas un grafā tām nav analogu.

Tad uzdevums sasaucas ar maksimālās plūsmas grafā atrašanu - varam ievērot, ka ir sadalošā kopa no divām virsotnēm, kur atbilstošajām salām vienlaikus cauri var doties divi kaķi, bet atlikušajiem kaķiem to šķērsošanai būs nepieciešams papildu laiks.

Bultiņas

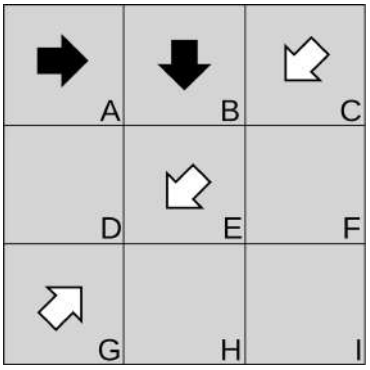
Ungārija

Tabulas rūtiņās ir vairākas bultiņas. Sākotnēji tās visas ir baltā krāsā.

Bultiņas ir jāizkrāso melnā krāsā tā, lai katra baltā bultiņa norādītu tikai uz vienu citu balto bultiņu, un katra melnā bultiņa norādītu uz divām citām melnām bultiņām.

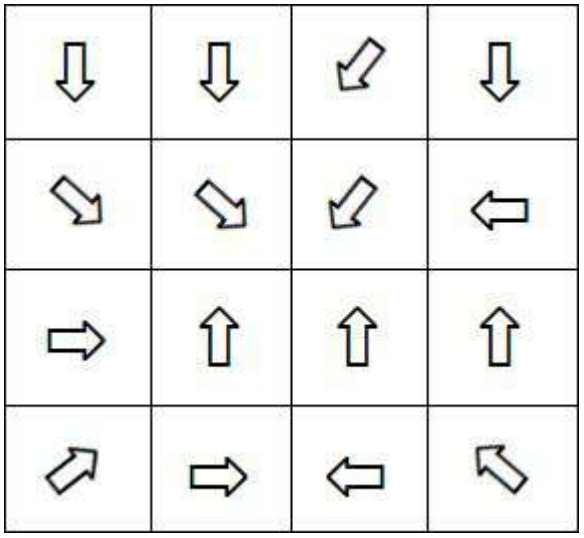
Piemēram, attēlā zemāk:

- Melnā bultiņa A norāda uz tieši vienu melnu bultiņu B un vienu balto bultiņu C.
- Baltā bultiņa C norāda uz tieši divām baltām bultiņām E un G.

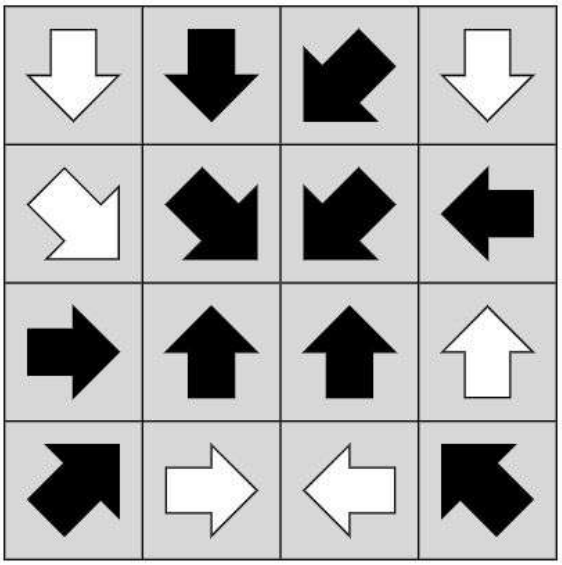


Uzdevums:

Izkrāso bultiņas tā, lai tas atbilstu uzdevuma nosacījumiem! Katra bultiņa var nomainīt krāsu no baltas uz melnu, kad uz tās uzklikšķina.



Pareizā atbilde: šis ir iespējamais risinājums:



Kā tas ir saistīts ar informātiku?

Lai atrastu atrisinājumu, dažreiz jūs varat sākt meklēšanu, to turpināt soli pa solim līdz nonākat strupceļā. Tad nepieciešams pakāpties soli atpakaļ un izmēģināt citu turpinājuma variantu. Šādu pieeju datorzinātnē sauc par atgriezmeklēšanu un tā noder tādos klasiskos uzdevumos kā astoņu šaha dāmu izvietošana, labirinta apstaigāšana vai mugursomas pakošana.

Tādas programmēšanas valodas kā Icon, Planner un Prolog izmanto atgriezmeklēšanu atbilžu ģenerēšanai.

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

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

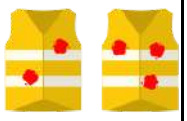
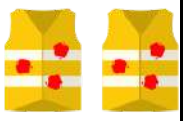
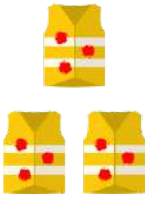
Netīrie formas tērpi

Dienvidkoreja

Tomātu festivālā tika notraipītas bebru formas, kuras tagad ir jāizmazgā vienā veļas mašīnā. Veļas mašīna vienā mazgāšanas ciklā var izmazgāt līdz trīs formām.

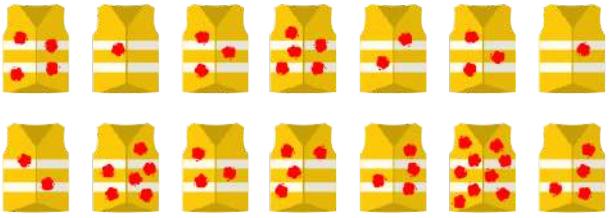
- Vienas formas mazgāšana aizņem tik stundas, cik pleķu ir uz formas;
- Divu formu mazgāšana aizņem tik stundas, cik pleķu ir uz netīrākās formas no divām;
- Trīs formu mazgāšana aizņem tik stundas, cik pleķu ir uz otras netīrākās formas no trim.

Zemāk redzamā tabula parāda variantus, kā formas iespējams izmazgāt triju stundu laikā:

1 forma	2 formas	2 formas	3 formas	3 formas
				

Uzdevums:

Kāds ir mazākais iespējamais laiks 14 zemāk attēlā redzamo formu izmazgāšanai?



Pareizā atbilde: 14 stundas

Šī uzdevuma mērķis ir aprēķināt mazāko nepieciešamo laiku 14 formu izmazgāšanai vienā veļas mašīnā, sagrupējot tās pa vienai, divām vai trijām katrā mazgāšanas ciklā.

Tā kā 14 nevar izdalīt ar 3, mums ir nepieciešami 4 cikli ar 3 formām un viens cikls ar 2 formām.

Lai samazinātu mazgāšanas laiku, formām ar vismazāk traipiem ir jābūt visnetīrākajām tajos ciklos, kuros tiek mazgātas 2 formas, bet otrām netīrākajām tajos ciklos, kuros tiek mazgātas 3 formas (jo šīs ir tās, kas nosaka cikla mazgāšanas ilgumu).

Tādējādi labākais veids, kā tās sadalīt, ir sagrupējot formas ar vismazāk traipiem tā, lai mēs iegūstam 5 pārus. Forma ar visvairāk traipiem katrā pārī noteiks cikla mazgāšanās ilgumu, tāpēc mēs varam pārējās formas izdalīt pa jau esošajiem pāriem, iegūstot labākā varianta ciklus.

Lai to izdarītu, formas ir nepieciešams sakārtot augošā secībā:
1 1 2 2 3 3 3 4 4 4 4 5 6 9

Tagad sagrupējam rindas pirmās divas formas kopā, tad otrās divas un tā tālāk, kamēr iegūstam šādus piecus pārus:
(1,1) (2, 2) (3, 3) (3, 4) (4, 4)

Ja mēs atlikušās formas iedalām izveidotajos pāros, iegūstam šādu risinājumu (ir iespējami arī citi risinājumi, kas izveido tādu pašu mazgāšanas reižu skaitu, kas ir iegūstami, izdalot atlikušās formas citādi):

(1, 1, 4) + (2, 2, 5) + (3, 3, 6) + (3, 4, 9) + (4, 4) = 1 + 2 + 3 + 4 + 4 = 14

Šādā veidā tiek iegūta atbilde 14.

Kā tas ir saistīts ar informātiku?

Optimizācijas uzdevumos ir nepieciešams starp vairākiem derīgiem atrisinājumiem atrast vislabāko. Šajā uzdevumā, lai atrastu labāko atrisinājumu, vispirms formas ir jāsakārto pēc tomātu traipu skaita. Pēc tam šī grupēšana ir jāatkārto, lai būtu iespējams minimizēt kopējo mazgāšanas laiku. Tādejādi tiks nodrošināts, ka atrisinājums, kas iegūts optimizācijas gaitā, vienmēr būs vislabākais iespējamais.

Viens no labi zināmiem optimālā risinājuma iegūšanas algoritmiem ir rijīgais algoritms. Šāda algoritma izpildes gaitā optimālais atrisinājums tiek būvēts soli pa solim. Bieži pirms rijīgā algoritma izpildes datus nepieciešams sakārtot, jo elementu izvēles secība var būt būtiska optimālā atrisinājuma iegūšanas sastāvdaļa. Kopumā (globāli) optimāls risinājums tiek veidots kā lokāli optimālu atrisinājumu virkne.

Rijīgie algoritmi ir vienkārši saprotami un realizējami, tie tiek izmantoti tādos klasiskos optimizācijas uzdevumos kā optimālas darbu secības noteikšana un īsākā ceļa atrašana grafā.

https://en.wikipedia.org/wiki/Sorting_algorithm
https://en.wikipedia.org/wiki/Greedy_algorithm
https://en.wikipedia.org/wiki/Bellman_equation#Bellman.27s_Principle_of_Optimality

Ūdens liešana

Holande

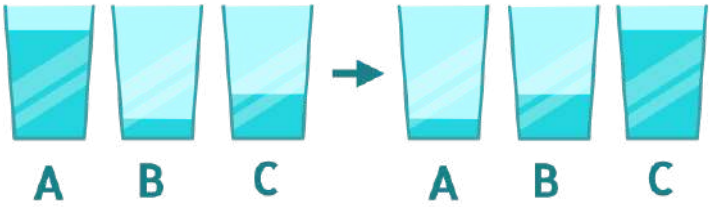
Trīs spaiņi A, B un C, ar dažāda daudzuma ūdeni ir novietoti vienā rindā. Neviens no šiem spaiņiem sākotnēji nav pilns.

Uz spaiņiem nav redzamas nekādas tilpuma atzīmes, tomēr salīdzināt ūdens daudzumu spaiņos ir iespējams.

Lai to izdarītu, atļauts izmantot tikai vienu vai vairākas no zemāk nosauktajām un uzskaitītajām darbībām (jāņem vērā, ka ne visas darbības ir vienmēr izpildāmas):

- "iztukšošana": paņem spaini un ielej visu tā saturu citā spainī;
- "izlīdzināšana": paņem spaini un izlej no tā tik daudz ūdens citā spainī, lai citā spainī atrastos tikpat daudz ūdens, cik trešajā spainī;
- "piepildīšana": paņem spaini un ielej tik daudz tajā esošā satura citā spainī, lai tas būtu pilns.

Uzdevums ir izmantot vienu vai vairākas no augstāk aprakstītajām darbībām, lai iegūtu tādu pašu spaiņu saturu, kā tas ir parādīts zemāk redzamajā attēlā, neizmantojot liekus spaiņus.



Pēc darbību izmantošanas, spainī A jābūt tikpat daudz ūdens, cik tas bija sākotnēji spainī B, spainim B būtu jābūt tikpat daudz ūdens, cik sākotnēji bija spainī C, un spainī C būtu jābūt tikpat daudz ūdens, cik sākotnēji bija spainī A.

Uzdevums:

Kurš no turpmākajiem apgalvojumiem ir patiess?

Atbilžu varianti:

A) Vēlamo iznākumu var panākt bez "iztukšošanas" darbības

B) Vēlamo iznākumu var panākt bez "izlīdzināšanas" darbības

C) Vēlamo iznākumu var panākt bez "piepildīšanas" darbības

D) Vēlamo iznākumu nevar panākt, izmantojot tikai šīs trīs darbības

Pareizā atbilde: D

Izpildot kādu no darbībām (iztukšošana, izlīdzināšana, piepildīšana) jebkādā skaitā, iegūst vienu no šiem iznākumiem:

- Viens no spaiņiem ir tukšs
- Divi no spaiņiem satur vienādu daudzumu ūdens

- Viens spainis ir pilns

Vēlamajam gala iznākumam nepiemīt neviens no šiem stāvokļiem, tāpēc pārveidošana nav iespējama.

Kā tas ir saistīts ar informātiku?

Trīs uzdevumā minētās operācijas ir saistītas ar informācijas zaudēšanu. Viens no paņēmieniem, kā noteikt, vai informācija tiek saglabāta operācijas izpildes laikā, ir pārbaudīt, vai tā ir apgriežama (iespēja iegūt precīzi tādu situāciju, kā bija pirms operācijas izpildes).

Šķidrumu daudzumu spaiņos sākumstāvoklī var attēlot ar trīs atšķirīgu skaitļu palīdzību, kur katrs no tiem ir lielāks par 0 (spainis nav tukšs) un mazāks par 1 (spainis nav pilns). Ja tiek veikta iztukšošanas darbība, tad kāds no skaitļiem kļūs 0, ja izlīdzināšana, tad divi skaitļi kļūs vienādi, bet, ja piepildīšana, tad viens no skaitļiem kļūs vienāds ar 1. Nevienā no šiem gadījumiem nav iespējams atjaunot situāciju pirms attiecīgās operācijas izpildes.

Programmētājiem vienmēr rūpīgi jāpiedomā par iespējamajiem programmu radītajiem efektiem un jāpārliecinās, ka būtiskā informācija vienmēr tiek saglabāta.

Klasisks informācijas saglabāšanas vingrinājums ir divu mainīgo vērtību apmainīšana vietām. Pieņemot, ka sākumā mainīgā a vērtība ir 0.5, bet mainīgā b vērtība ir 0.2, naivais algoritms

```
a := b;
b := a;
```

ir kļūdains, jo pēc šo operaciju izpildes gan a, gan b vērtība ir 0.2, bet sākotnējā vērtība 0.5 ir neatgriezeniski zaudēta. Klasisks pareizs risinājums izmanto papildu mainīgo:

```
c := b;
b := a;
a := c;
```

Asprātīga alternatīva, neizmantojot papildu mainīgo, ir šāda:

```
a := a – b;
b := b + a;
a := b – a;
```

Šajā gadījumā tiek saglabāta sākotnējo vērtību starpība un tas ļauj aprēķināt sākotnējās vērtības otrajā un trešajā programmas koda rindā.

Dotajā uzdevumā katra no trim atļautajām operācijām ir saistīta ar informācijas zaudēšanu un nav pieejami arī papildu mainīgā analogs - lieki spaiņi.

Informācijas saglabāšana operāciju laikā ir ļoti svarīga tādā jaunā datorzinātņu jomā kā kvantu skaitļošana, kur visām operācijām, lai tās tiktu uzskatītas par korektām, jābūt apgriežamām.

Atsauces:

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

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