

Bebr[a]s

'25

2025. gada 1. kārtas uzdevumi ar atbildēm

9.-10. klase

INFORMATĪVIE ATBALSTĪTĀJI



Izglītības un zinātnes
ministrija

start(it)

www.startit.lv

Satura rādītājs

Pārvadāšanas stratēģija

Autobuss

Trešais

Stāvvietā

Aizraujošais nobrauciens

Pulksteņa segments

Bebrusalas biļetens

Būla figūras

Puku stādīšana

Milti

Akmens, šķēres, papīrīt's un maiņas

Printera tīrīšana

Ciparu mikslis

Priecīgā aplī

Pārvadāšanas stratēģija

Azerbaidžāna

Bebram jāpaņem pieci apslimuši eksotisku sugu kaķi, lai tos nogādātu pie veterinārārsta. Viņam ir vairāki dzīvnieku pārvadāšanas konteineri. Katram konteineram ir pieļautā kopējā svara ierobežojums kilogramos.

Konteiners	Pieļaujamā svara ierobežojums kilogramos
A	10
B	15
C	20
D	5

Bebra mērķis ir pārvest visus kaķus, izmantot pēc iespējas mazāku konteineru skaitu, nevienam nepārsniedzot atļautā svara ierobežojumu. Katram kaķim ir zināms tā svars:

Kaķa vārds	Svars kilogramos
<i>Tiny</i>	3
<i>Blacky</i>	7
<i>Fatboy</i>	10
<i>Colorina</i>	5
<i>Lazyboy</i>	6

Jautājums / Izaicinājums

Kura no tālāk minētajām iespējām ļaus pārvadāt visus kaķus, izmantojot pēc iespējas mazāku konteineru skaitu?

Atbilžu varianti / Interaktivitātes apraksts

a) *Tiny* un *Blacky* konteinerā B; *Fatboy*, *Colorina* un *Lazyboy* konteinerā C.

b) *Tiny* un *Blacky* konteinerā A, *Fatboy* konteinerā B, *Colorina* konteinerā D, *Lazyboy* konteinerā C.

c) *Tiny* un *Colorina* konteinerā A, *Blacky* un *Lazyboy* konteinerā B, *Fatboy* konteinerā C.

d) *Tiny*, *Blacky* un *Fatboy* konteinerā C, *Colorina* konteinerā D, *Lazyboy* konteinerā A.

e) *Fatboy* un *Colorina* konteinerā B, *Tiny*, *Blacky* un *Lazyboy* konteinerā C.

Atbildes skaidrojums

Pareizā atbilde ir e variants: *Fatboy* (10 kg) + *Colorina* (5 kg) = 15 kg ietilpst konteinerā B (15 kg) *Tiny* (3 kg) + *Blacky* (7 kg) + *Lazyboy* (6 kg) = 16 kg ietilpst konteinerā C (20 kg). Lai ātri atrastu pareizo atbildi, varat saskaitīt katrā variantā izmantoto konteineru skaitu. a un e variantā tiek izmantoti divi konteineri. c un d variantā tiek izmantoti trīs konteineri. b variantā tiek izmantoti četri konteineri. Tātad, jūs sākat, pārbaudot a un e variantus.

Izrādās, ka a variants nav iespējams, jo trīs kaķi kopā svērtu 21 kg. Tas pārsniedz konteineru C ietilpību (20 kg). Tātad paliek tikai e atbilde, un tā atbilst svara ierobežojumiem.

Šī ir informātika

Šis uzdevums modelē resursu sadali ar ierobežojumiem, kas ir informātikas pamatkonceptija. Tas atgādina mugursomas problēmas variāciju, kombinatorisku optimizāciju, kur ir dots priekšmetu skaits un to svars, kā arī noliktava ar svara ierobežojumu; risinājums maksimāli palielinās to priekšmetu skaitu, kurus var uzglabāt, nepārsniedzot svara ierobežojumu.

Šajā gadījumā kaķi ir priekšmeti, un krātuvi pārstāv konteineri:

- katram konteineram ir svara ierobežojums. Mērķis ir samazināt izmantoto konteineru skaitu.

Studentiem jānovērtē derīgas kombinācijas, lai optimizētu izmantotos resursus, kas ir galvenā ideja operētājsistēmās, atmiņas pārvaldībā un failu sistēmās.

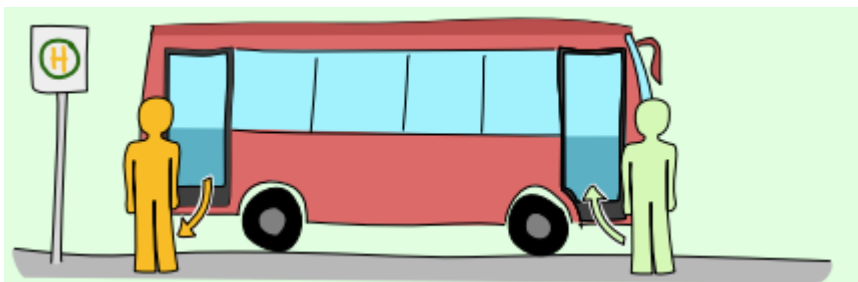
Šī ir skaitļošanas domāšana.

Uzdevums ietver dekompozīciju, abstrakciju, kombināciju sistemātisku testēšanu un risinājumu salīdzināšana optimalitātes noteikšanai (izmantots vismazāk konteineru).

Autobuss

Beļģija

Starppilsētu autobuss bez vadītāja kursē pa noteiktu maršrutu. Parasti autobuss apstājas pieturā, kad cilvēki gaida iekāpšanu vai kad pasažieri vēlas izkāpt no autobusa. Taču šobrīd autobusa dators nedarbojas pareizi, un autobuss apstājas pieturā tikai tad, kad ir pasažieri, kas gaida iekāpšanu, un vienlaikus ir pasažieri, kas vēlas izkāpt no autobusa. Autobuss vienmēr apstājas galapunktos (pirmajā un pēdējā pieturā), neskatoties uz aprakstīto datora kļūmi.



Autobusu pieturas ir numurētas no 1 līdz 5. Tajās gaida šādi pasažieri: -

- 1. pieturā Marija vēlas iekāpt un viņa vēlas izkāpt
- 2. pieturā -
- 1. pieturā Vasils vēlas iekāpt un viņš vēlas izkāpt
- 4. pieturā -
- 2. pieturā Simons vēlas iekāpt un viņš vēlas izkāpt
- 5. pieturā -
- 3. pieturā Jeļena vēlas iekāpt un viņa vēlas izkāpt
- 4. pieturā -
- 3. pieturā Kalina vēlas iekāpt un viņa vēlas izkāpt
- 5. pieturā.

Jautājums / Izaicinājums

Kāds ir visu to pasažieru saraksts, kuri nevarēs nokļūt savā vēlamajā pieturā?

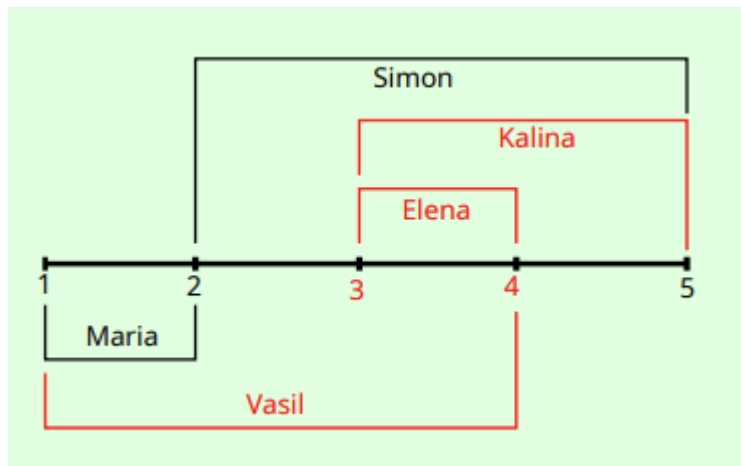
Atbilžu varianti / Interaktivitātes apraksts

- A. Jeļena
- B. Vasils un Kalina
- C. Vasils, Jeļena un Kalina
- D. Vasils

Atbildes skaidrojums

Pareizā atbilde: C

Autobuss neapstāsies 3. un 4. pieturā, jo, vai nu tur negaida cilvēki, vai arī nav pasažieru, kuri tur vēlas izkāpt. Tomēr autobuss pieturēs 1., 2. un 5. pieturā. Tādējādi divas personas, kas gaida 3. pieturā, Jeļena un Kalina, nevarēs iekāpt un tāpēc nerasniegs savas vēlamās pieturas. Vasilis iekāpj 1. pieturā un vēlas izkāpt 4. pieturā, bet autobuss tur neapstājas. Tādējādi Vasilis, Jeļena un Kalina nerasniegs savas vēlamās pieturas.



Interaktīvs jautājums/izaicinājums

Atzīmējiet visus pasažierus, kuri var nokļūt savā vēlamajā pieturā?

Atbilde:

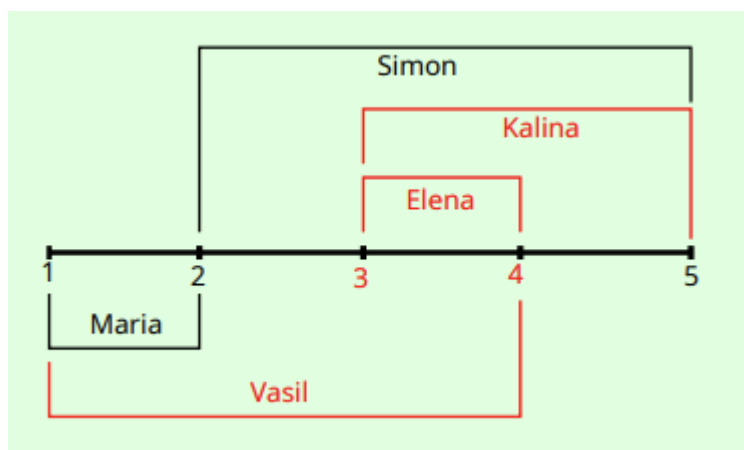
Marija un Saimons

Atbildes skaidrojums

Tikai Marija un Saimons nokļūs savās vēlamajās pieturās.

Autobuss neapstāsies 3. un 4. pieturā, jo, vai nu tur nav cilvēku, kas gaida, vai arī nav pasažieru, kuri tajās vēlētos izkāpt. Tomēr autobuss pieturēs 1., 2. un 5. pieturā.

Tādējādi divas personas, kas gaida 3. pieturā, Jeļena un Kalina, nevarēs iekāpt un tāpēc nerasniegs savas vēlamās pieturas. Vasilis iekāpj 1. pieturā un vēlas izkāpt 4. pieturā, bet autobuss tur neapstājas. Tādējādi Vasilis, Jeļena un Kalina nerasniegs savas vēlamās pieturas. To ilustratīvi parāda arī sarkanās līnijas nākamajā diagrammā. Izslēgšanas rezultātā atlikušie pasažieri Marija un Simons nokļūs savās vēlamajās pieturās. To ilustratīvi parāda arī melnās līnijas nākamajā attēlā.



Alternatīvs skaidrojums: No uzdevuma apraksta autobuss apstāsies 1., 2. un 5. pieturā, jo autobuss vienmēr pieturēs 1. un 5. pieturā, un 2. pietura ir vienīgā, kurā ir pasažieris, kas vēlas iekāpt un izkāpt. Tādējādi mums jāatzīmē tikai pasažieri, kas vēlas iekāpt un izkāpt 1., 2. un 5. pieturā, un nav jāņem vērā pasažieri, kas vēlas iekāpt vai izkāpt 3. un 4. pieturā. Tam atbilst tikai Marija (1. un 2. pietura) un Simons (2. un 5. pietura). To ilustratīvi parāda arī melnās līnijas iepriekš redzamajā diagrammā.

Šī ir informātika

Uzdevums prasa atpazīt un ņemt vērā izmaiņas apstāšanās shēmā. Parastais modelis ir "apstāties, ja kāds iekāpj VAI izkāpj". Izmainītais (bojātais) modelis ir "apstāties, ja kāds iekāpj UN izkāpj". Pirmā un pēdējā pietura attēlo fiksētu, beznosacījumu modeli. Problēmas izpratne ietver noteikumu vai nosacījumu kopumu, ko ievēro autobusa dators. Bojājums ievieš konkrētu, mainītu noteikumu. Šis uzdevums ir vienkārša reālās pasaules transporta sistēmas modelēšana ar konkrētiem noteikumiem, kas regulē tās darbību. Lēmums apstāties ir atkarīgs no datiem (vai cilvēki gaida un vai pasažieri vēlas izkāpt) un loģiskajiem nosacījumiem, kas tiek piemēroti šiem datiem. Datu apstrāde un loģiskās operācijas ir informātikas centrā.

Šī ir skaitļošanas domāšana

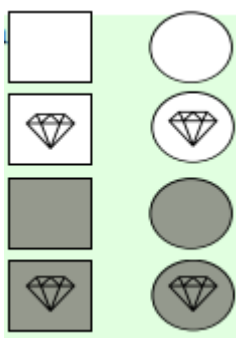
Visām izmaiņām, kas notiek laika gaitā, ir jāseko līdzi. Datordomāšanā tā ir tipiska notikumu izsekošana ciklā un izmaiņu veikšana sarakstos atkarībā no loģisko nosacījumu izpildes. Identificējamais modelis ir tas, vai pieturā ir gan pasažieri, kas vēlas iekāpt, GAN izkāpt, kā arī triviāli sākuma un beigu pieturu robežgadījumi.

Trešais

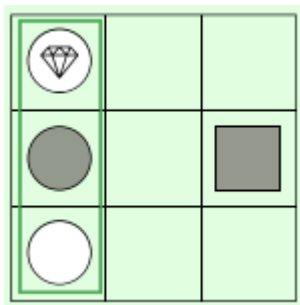
Brazīlija

Bebri Anna un Bento spēlē spēli ar nosaukumu "Trešais". Viņi izmanto 3x3 rūtiņas lielu spēles laukumu un astoņas unikālas figūriņas ar trim raksturīgām īpašībām:

- Krāsa: pelēka vai balta.
- Forma: kvadrāts vai aplis.
- Marķējums: ar vai bez romba.

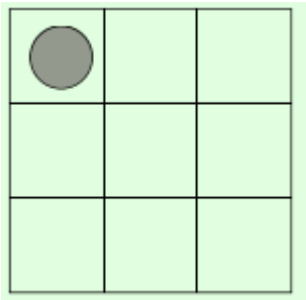


Lai uzvarētu, spēlētājam jāaizpilda laukuma rinda, kolonna vai diagonāle, kur trim figūriņām ir vismaz viena kopīga iepriekš aprakstītā īpašība. Piemēram:

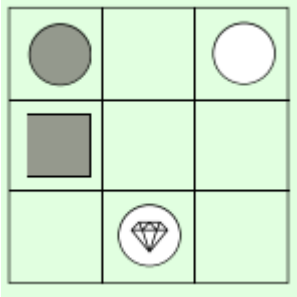


Iepriekš redzamajā spēles laukumā viens spēlētājs ir uzvarējis, jo novietojis trešo figūriņu pirmajā kolonnā, iegūstot trīs figūriņas ar vienādu raksturlielumu vienā līnijā: apla formas figūriņas vienā kolonnā.

Spēle sākas ar tukšu spēles laukumu. Spēlētāji gājienus izdara pēc kārtas, bet pretinieks izvēlas figūriņu, kas jānovieto. Pēc figūriņas novietošanas spēlētājs izvēlas nākamo figūriņu, kas laukumā jānovieto pretiniekam utt. Anna sāka spēli, izvēloties pelēko apli bez romba (), lai Bento spēlētu, un pēc tam Bento to novieto uz spēles laukuma.



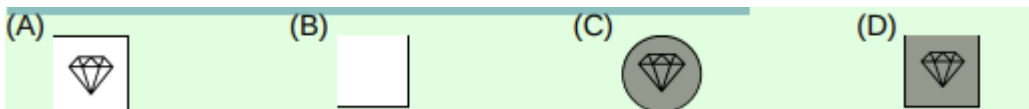
Viņi turpināja pārmaiņus, līdz spēles laukumā iestājās šāda situācija:



Jautājums / Izaicinājums

Kuru no tālāk norādītajām figūrām Anna var dot Bento novietot uz galda, lai nodrošinātu, ka viņš NEuzvar savā gājienā?

Atbilžu varianti / Interaktivitātes apraksts

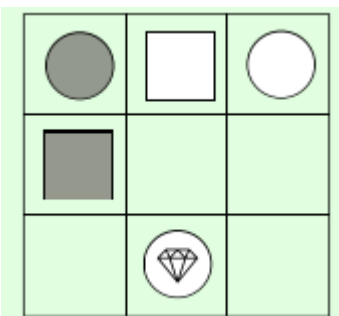


Atbildes skaidrojums

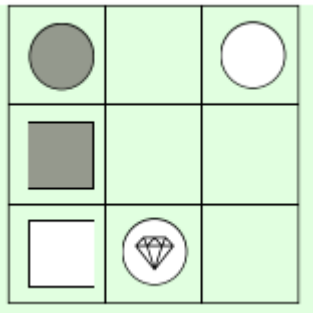
Pareizais variants ir (A).

Novietojot figūru no (A) iespējas, tiek nodrošināts, ka Bento nevarēs novietot trīs figūras ar kopīgu īpašību, tādējādi neļaujot viņam uzvarēt. (B) iespēja ir nepareiza. Ir divas iespējamās vietas, kur Bento varētu novietot šo figūru un uzvarēt:

- Viņš var novietot trīs figūras bez romba pirmajā rindā.

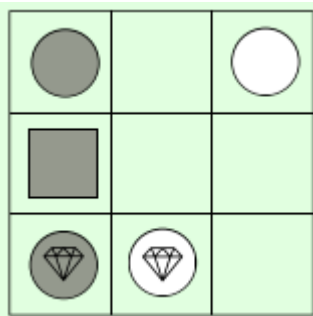


- Viņš var novietot trīs figūras bez romba pirmajā kolonnā.

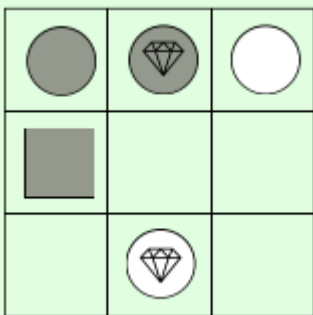


(C) iespēja ir nepareiza. Ir divas iespējamās vietas, kur Bento varētu novietot šo figūru un uzvarēt:

- Viņš var novietot trīs pelēkas figūras pirmajā kolonnā.

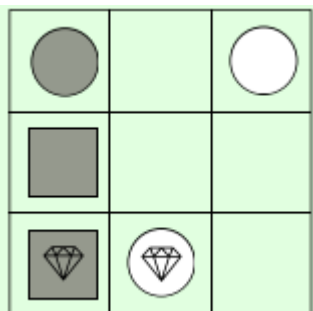


- Viņš var novietot trīs figūras ar apla formu pirmajā rindā.



(D) iespēja ir nepareiza. Ir viena iespējama vieta, kur Bento varētu novietot šo figūriņu un uzvarēt:

- Viņš var novietot trīs pelēkas figūriņas pirmajā kolonnā.



Šī ir informātika

Atribūtu jēdziens datorzinātnēs ir saistīts ar objekta raksturlielumu vai īpašību. Piemēram, objektorientētās programmēšanas valodās atribūti ir dati, kas apraksta objekta stāvokli (piemēram, krāsa, forma, izmērs utt.). Spēlē šī ideja izpaužas ļoti skaidri: katrai figūriņai spēlē ir trīs binārie atribūti, ir $2^3=8$ unikālas figūriņas, katrai ar atšķirīgu šo atribūtu kombināciju. Spēles mērķis ir novietot trīs figūriņas, kurām ir viens un tas pats atribūts, taisnā līnijā neatkarīgi no tā, kurš tās novietoja. Un, lai uzvarētu šajā spēlē, jums ir jāanalizē katrs gājieni un jāparedz pretinieka iespējamie atbildes gājieni. Tas nozīmē, ka jums ir jāizvērtē potenciālie gājieni un to sekas, lai izdarītu labāko izvēli. Tas padara spēli ļoti piemērotu programmēšanai, izmantojot lēmumu kokus, piemēram, "ja-citādi" paziņojumus. Turklāt spēlētājiem ir jāievēro spēles noteikumi. Noteikumi ir ļoti svarīgi ne tikai šajā spēlē, bet arī datu apstrādē datorsistēmās. Tie palīdz definēt attēlu failu, kredītkaršu numuru un daudzu citu strukturētu datu tipu formātus.

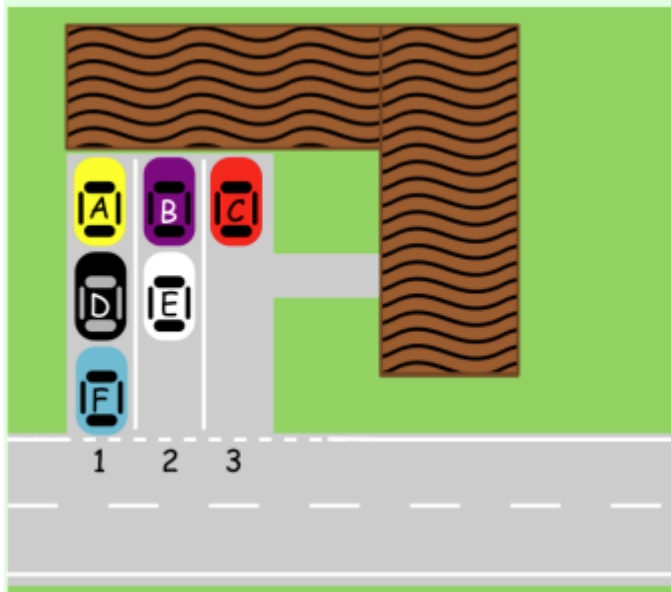
Šī ir skaitļošanas domāšana.

Lai atrisinātu šo uzdevumu, skolēniem ir jāizvērtē dažādi gājieni un jāparedz pretinieka iespējamās atbildes, lai katrā kārtā pieņemtu labāko lēmumu (Novērtēšana). Turklāt figūru pozīciju modeļu atpazīšana palielina viņu izredzes uzvarēt un palīdz novērst negaidītus pārsteigumus no pretinieka puses (Modeļu atpazīšana).

Stāvvietā

Slovēnija

Aija organizē dzimšanas dienas ballīti uz kuru uzaicinātie draugi ierodas ar automašīnām. Aijas piebraucamajā ceļā ir vieta deviņām automašīnām, kas jānovieto trīs kolonnās pa trim automašīnām viena pēc otras. Katram viesim tiek lūgts novietot automašīnu pirmajā brīvajā vietā jebkurā kolonnā.



Šajā piemērā automašīna A ieradās pirmā, automašīna D - otrā, automašīna C - trešā, un automašīnai F jāaizbrauc, pirms to var izdarīt automašīna D.

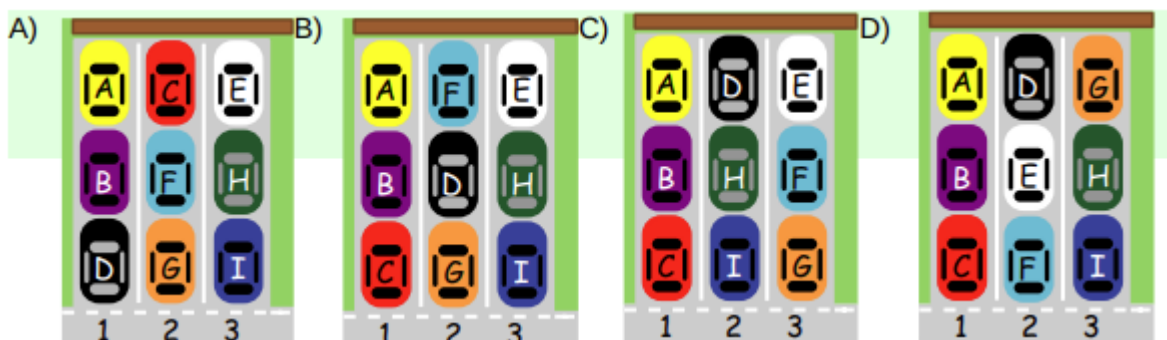
Viesi ierodas šādā secībā: Anna, Bobs, Cilda, Deivids, Elena, Frenks, Georgs, Heilija, Ivans.

Viņi pametīs ballīti šādā secībā: Georgs, Deivids, Frenks, Ivans, Heilija, Cilda, Bobs, Anna, Elena.

Jautājums / Izaicinājums

Sakārtojiet automašīnas kolonnās viesu ierašanās secībā tā, lai neviena automašīna neaizšķērsotu automašīnu, kurai jāaizbrauc agrāk. Izvēlieties pareizo variantu.

Atbilžu varianti / Interaktivitātes apraksts



Atbildes skaidrojums

Pareizā atbilde ir A.

Mēs varam analizēt situāciju katram viesu pārim:

- Anna ierodas pirms Cildas, un Cilda dodas prom pirms Annas. Tātad viņas var novietot automašīnas vienā kolonnā. Tas pats attiecas uz Annu un Bobu, kā arī uz Bobu un Cildu.

- Anna un Elena ierodas tādā pašā secībā, kādā viņi aizbrauc, tāpēc viņi nevar novietot automašīnas vienā kolonnā. Tas pats attiecas uz Bobu un Elenu; Cildu un Elenu; Deividu pret Elenu, Frenku, Heiliju un Ivanu; Frenku attiecībā pret Heiliju un Ivanu; Georgu pret Heiliju un Ivanu.

Tātad Anna, Bobs un Cilda visi var novietot automašīnu 1. kolonnā savā ierašanās secībā. Kad ierodas Deivids, lai viņš novieto automašīnu 2. kolonnā, jo 1. kolonna ir pilna. Kad ierodas Elena, viņai jānovieto automašīna citā kolonnā nekā Deivida automašīna, tāpēc viņai jānovieto automašīna 3. kolonnā. Tad ierodas Frenks, arī viņam jānovieto automašīna citā kolonnā nekā Deivida automašīna, tāpēc lai viņš novieto automašīnu 3. kolonnā, kas ir labi, jo viņš aizbrauks pirms Elenas, kura tur ir novietojusi automašīnu. Kad Georgs ierodas, tā kā viņš ir pirmais, kas dosies prom, viņam jānovieto kā pēdējam kolonnā, tāpēc ļaujiet viņam novietot automašīnu kā trešajai automašīnai 3. kolonnā. Šajā brīdī mums ir divas vietas 2. kolonnā. Ierodas Heilija, kam seko Ivans, un Ivans aizbrauks pirms Heilijas, tāpēc viņi var novietot automašīnu 2. kolonnā ierašanās secībā. Pēc ballītes draugi aizbrauc šādā secībā: Georgs, Deivids, Frenks, Ivans, Heilija, Cilda, Bobs, Anna, Elēna. Nevienu neaizšķērso kāda cita automašīna, jo:

1. Vispirms aizbrauc Deivids un Georgs.
2. Kad Georgs aizbrauc, Frenka automašīna tiek atbloķēta.
3. Deivids atbloķē Boba automašīnu.
4. Frenks atbloķē Cildas automašīnu.
5. Ivans atbloķē Heilijas automašīnu.
6. Heilija atbloķē Elēnas automašīnu.

7. Cilda iztukšo 2. kolonnu.
8. Bobs atbloķē Annas automašīnu.
9. Anna un Elēna var aizbraukt, jo viņu automašīnas ir vienīgās viņu kolonnās.

Atbilde B ir nepareiza, jo Deivids ierodas pirms Frenka, tāpēc viņš nevar novietot automašīnu aiz viņa.

Atbilde C ir nepareiza, jo Deividam jāaizbrauc pirms Heilijas, bet Heilija ir novietojusi automašīnu aiz viņa.

Atbilde D ir nepareiza, jo, lai gan viesu automašīnas ir novietotas to ierašanās secībā, Georga automašīna ir bloķēta, un viņam jāaizbrauc pirmajam.

Šī ir informātika

Automašīnu novietošanas kolonnās Aijas piebraucamajā ceļā atbilst tam, ko informātikā sauc par steku. Steks ir datu struktūra, ko izmanto, lai glabātu secīgas datu vai objektu kolekcijas. Datu struktūras ir svarīgas informātikā, jo tās ļauj mums organizēt datus tā, lai tos varētu efektīvi apstrādāt, uzglabāt un no tām izgūt datus. Steks darbojas kā trauku kaudze: kad ir jāuzglabā jauns objekts, tas tiek novietots steka augšpusē. Vienmēr var tikai paskatīties (ielūkoties) vai noņemt (izņemt) objektu no steka augšas. Datu struktūru ar šāda veida uzvedību sauc arī par "pēdējais iekšā, pirmais ārā". Tātad, izmantojot steku, var tieši piekļūt un noņemt tikai pēdējo stekam pievienoto elementu, tāpat kā automašīnas šajā uzdevumā. Problēma, kas saistīta ar daudzu automašīnu izvietošanu vairākos stekos, lai tās nebloķētu viena otru, kad tām ir jāaizbrauc, un līdzīgas problēmas, tiek sauktas par vairāku steku problēmām, un algoritmi to risināšanai tiek izmantoti resursu piešķiršanā uzdevumiem, piemēram, daudzkodolu procesoru ierīcēs, kur var notikt paralēla apstrāde.

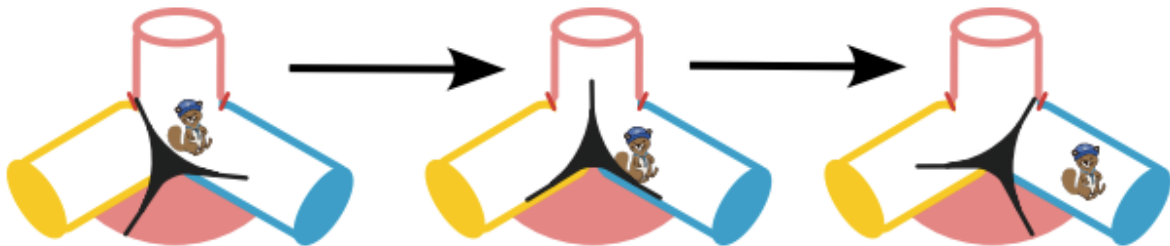
Šī ir skaitļošanas domāšana

Skolēniem ir jāseko automašīnu ierašanās secībai un jāpārbauda, kur tās var novietot, un pēc tam jāpārbauda arī, kad katra automašīna varēs atstāt savu stāvvietu. Ar šī uzdevuma palīdzību viņi attīsta algoritmisko domāšanu un loģisko spriešanu.

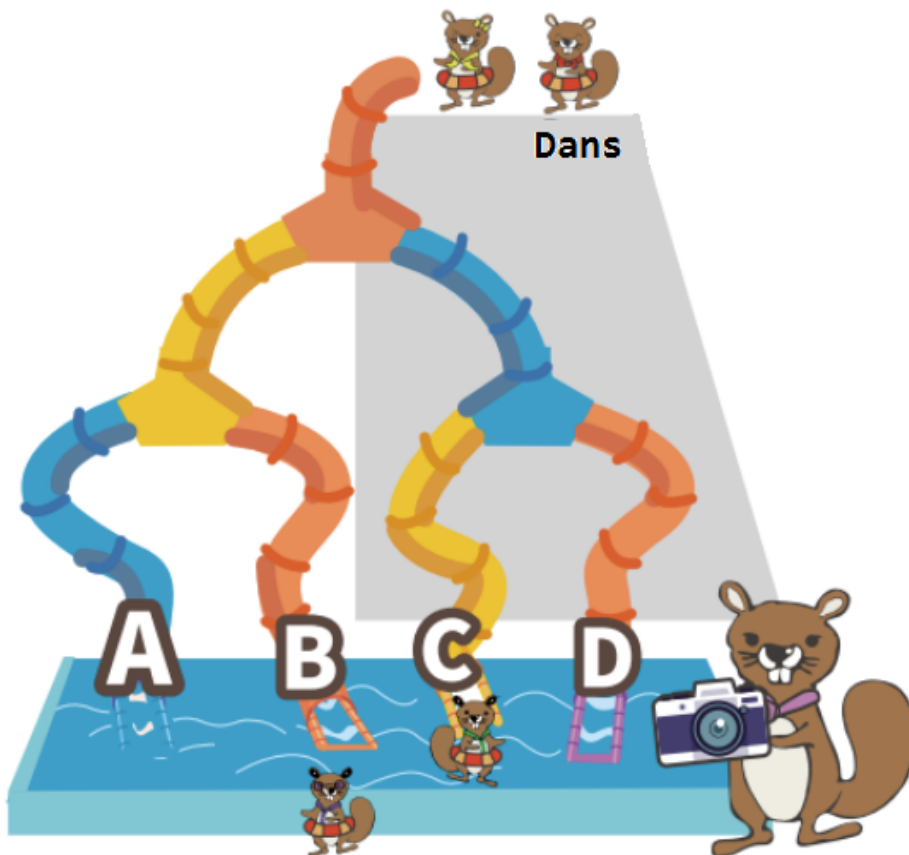
Aizraujošais nobrauciens

Taivāna

Bebru ūdens atrakciju parkā ir aizraujoša dinamisku ūdens slidkalniņu kaskāde, kas beidzas ar baseinu. Šīs diagrammas parāda, kā tas darbojas: mehānisms kontrolē katra slidkalniņa sazarojuma izejas virzienu; ikreiz, kad bebrs izslīd cauri atzarojumam, izejas virziens mainās.



Kā redzams nākamajā attēlā, mazais bebrs Dans vēlas izmēģināt šos ūdens slidkalniņus. Bebru mamma vēlas zināt, pa kuru slidkalniņu Dans iekritīs baseinā, lai varētu uzņemt labas bildes. Pirmais bebru iekrīt baseinā no B slidkalniņa, un pēc tam nākamais - no C slidkalniņa. Rindā uz nobraucienu gaida vēl viens bebrs un Dans būs nākamais pēc tā.



Jautājums / Izaicinājums

No kura slidkalniņa mazais bebrš Dans iekritīs baseinā?

Atbilžu varianti / Interaktivitātes apraksts

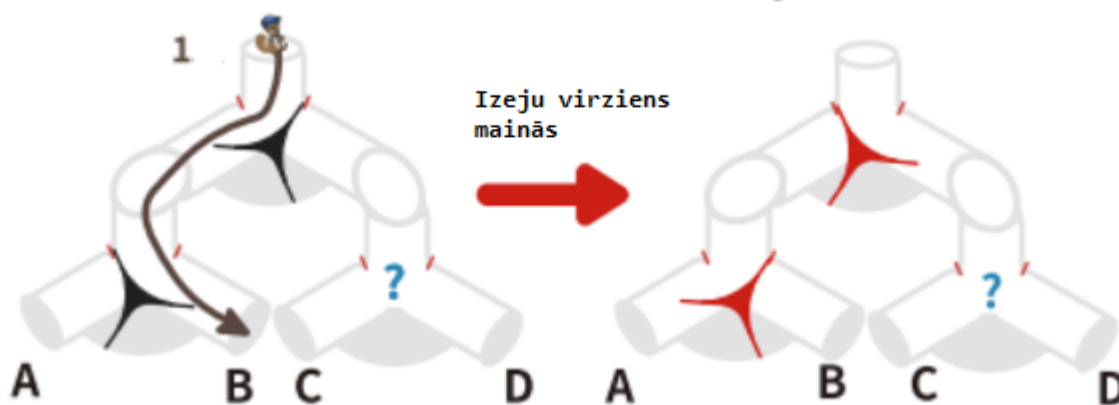
- A. No A slidkalniņa
- B. No B slidkalniņa
- C. No C slidkalniņa
- D. No D slidkalniņa

Atbildes skaidrojums

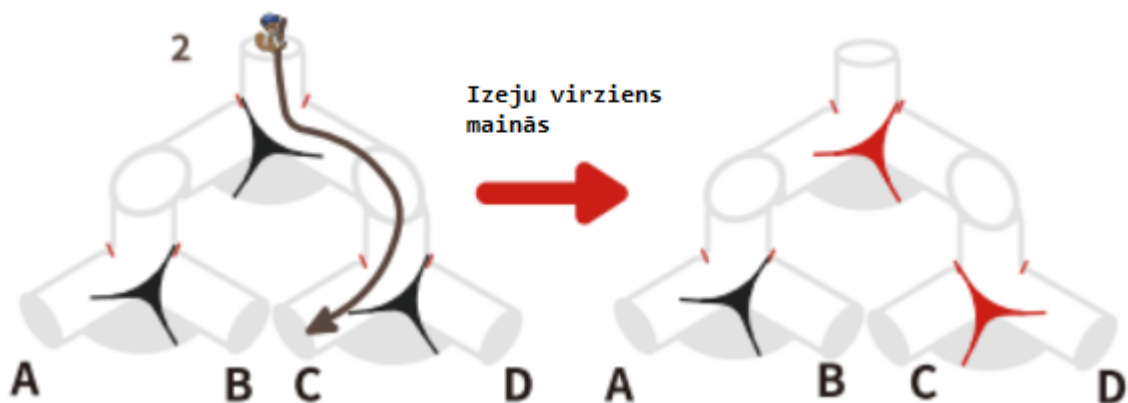
Atbilde ir D.

Tā kā izejas virziens mainās katru reizi, kad bebrš izslīd cauri atzarojumam, mēs varam noteikt atzarojuma izejas virzienu pēc tam, kad pirmie divi bebru ir izslīdējuši cauri.

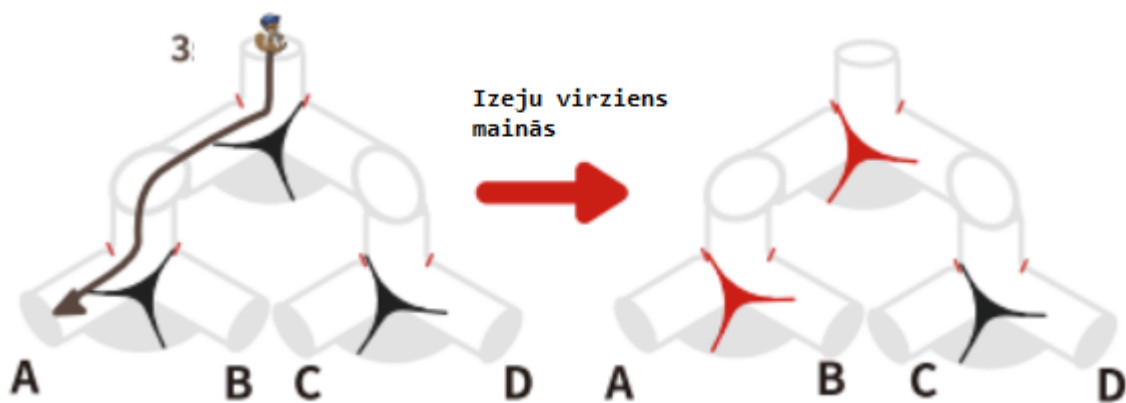
Kā redzams zemāk esošajā attēlā, pirmais bebrš iekrīt baseinā no B slidkalniņa; Mēs varam secināt, ka bebrš pirmajā atzarojumā iet pa kreisi un otrajā pa labi. Pēc tam šo divu atzarojumu izejas virziens mainās: pirmais atzarojums tiek atvērts pa labi, bet otrais (apakšējais kreisais) atzarojums - pa kreisi. Šajā brīdī mēs joprojām neko nezīnām par apakšējā labā atzarojuma izejas virzienu.



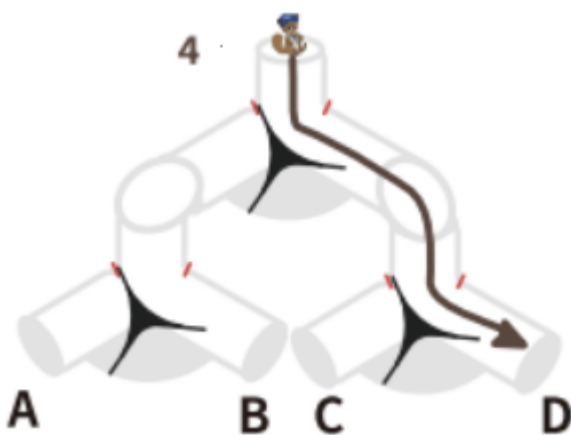
Līdzīgi, kā parādīts attēlā zemāk, nākamais bebrš iekrīt baseinā no C slidkalniņa; mēs varam secināt, ka apakšējā labā atzarojuma izejas virziens bija pa kreisi pirms bebra izslīdēšanas un pēc tam pagriežas pa labi.



Šajā brīdī mēs esam noskaidrojuši visu atzarojumu izejas virzienus, tāpēc varam izsekot maršrutu bebram pirms Dana, un atzarojumu izejas virzieniem pēc tam, kā parādīts attēlā zemāk.



Visbeidzot, mēs zinām, ka Dans slīdēs pa maršrutu, kas parādīts attēlā zemāk, un iekritīs baseinā pa D slidkalniņu.



Šī ir informātika

Šajā uzdevumā katra atzarojuma izejas virziens nosaka, pa kuru slidkalniņu bebrs slīdēs tālāk. Tas ir līdzīgi tam, kā programmēšanā darbojas nosacījuma instrukcijas. Tipisks

nosacījums atbilst šādam modelim: "Ja nosacījums ir patiess, dariet to; pretējā gadījumā dariet ko citu." Šajā uzdevumā mēs varam aprakstīt atzarojuma mehānisma darbību šādi: "Ja tas ir vērsts pa kreisi, bebrs tam slīdēs cauri pa kreiso slidkalniņu un mehānisms pārslēgsies uz labo pusi; pretējā gadījumā bebrs slīdēs cauri pa labi un mehānisms pārslēgsies uz kreiso pusi."

Kad bebrs izslīd cauri katram atzarojuma mehānismam, tas pats process atkārtojas. Bebrs seko mehānisma pašreizējam virzienam, un mehānisms pārslēdzas pretējā virzienā. Tāpēc katram mehānismam ir jāatceras patreizējais virziens, kas ir informācijas daļa, ko programmēšanā var attēlot kā mainīgo. Piemēram, mēs varam izmantot 0, lai attēlotu "pa kreisi", un 1, lai attēlotu "pa labi". Kad bebrs pārvietojas pa struktūru, programma varētu pārbaudīt katra mainīgā vērtību (mehānisma virzienu), izlemt, kuru ceļu izvēlēties, un atjaunināt mainīgo, lai atspoguļotu jauno stāvokli.

Šis mehānisms ir līdzīgs arī trigeru digitālajās shēmās — pamatvienībai, kas glabā vienu datu bitu un pārslēdzas starp diviem stāvokļiem. Katrs atzarojums uzvedas kā trigeris, mainot savu stāvokli katru reizi, kad tas tiek izmantots.

Lai noskaidrotu, kur bebrs nonāks, mēs soli pa solim simulējam katru darbību. Tas ir līdzīgi koda izsekošanai programmēšanā, kur programmas loģika tiek sekota rindiņai pa rindiņai, lai redzētu, kādu rezultātu tā rada.

Šī ir skaitļošanas domāšana

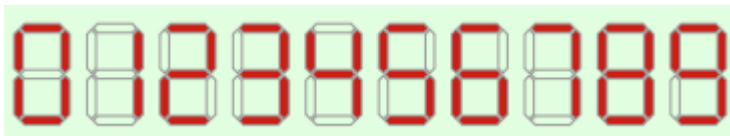
Studentiem ir jāpielieto loģiskā spriešana, lai izsekotu izmaiņām atzarojuma virzienos un precīzi prognozētu rezultātus. To darot, viņiem jāignorē stāsta elementi, piemēram, kas ir bebrs vai kā izskatās augļu dārzs, un tā vietā jākoncentrējas uz uzdevuma būtisko struktūru: katra atzarojuma secību un stāvokli. Šis vienkāršošanas process, koncentrējoties tikai uz būtiskajiem aspektiem, ir pazīstams kā abstrakcija.

Pulksteņa segments

Čehija



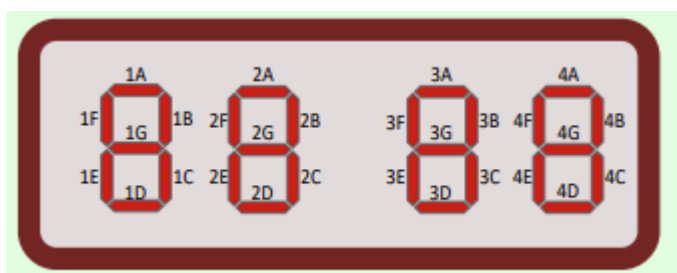
Standarta digitālais pulkstenis, kura rādījums mainās ik minūti, rāda laiku, izmantojot četrus ciparus. Katru ciparu attēlo septiņi gaismas segmenti. Lūk, kā, izmantojot segmentus, var attēlot skaitļus no 0 līdz 9:



Segmenti noliektos katru reizi, kad tie tiek aktivizēti (pārslēgti no izslēgta stāvokļa uz ieslēgtu). Pēc kāda laika kā pirmais būs jānomaina segments, kas tiek aktivizēts visvairāk reižu.

Jautājums / Izaicinājums

Kurš no 28 pulksteņa segmentiem ir jānomaina vispirms? Norādiet segmenta kodu (cipars, kam seko burts, skatīt attēlu).



Interaktivitātes apraksts:



Kurš no 28 segmentiem ir jānomaina vispirms? Atzīmējiet to, noklikšķinot uz tā.

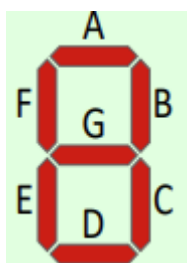
Katrs no 28 interaktīvajiem segmentiem tiek iezīmēts pēc noklikšķināšanas uz tā. Vienlaikus var atzīmēt tikai vienu segmentu.

Atbildes skaidrojums

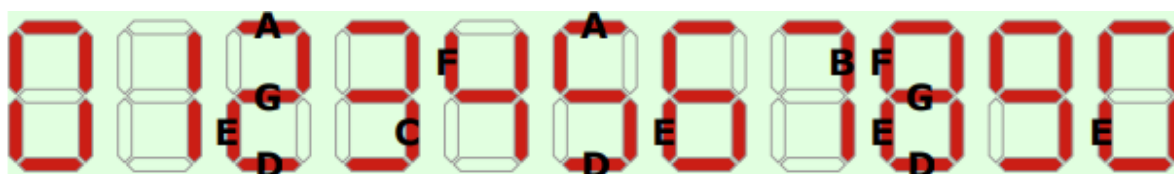


Pareizā atbilde ir: 4E (interaktīvā versija):

Katrs cipars displejā apzīmē atšķirīgu laika periodu: desmitus stundu, stundas, desmitus minūšu un minūtes. Cipars visvairāk pa labi, t. i., ceturtais cipars jeb cipars ar kodu 4, rāda minūšu izmaiņas. Pirms jebkura cita cipara izmaiņām mainīsies visi ceturta cipara segmenti. Visvairāk nolietotais segments būs ceturtajā ciparā, un tā segmenta kods ir E.



Mēs apskatīsim visus septiņus segmentus ceturtajā ciparā un noskaidrosim, cik reižu tie tiks aktivizēti (ieslēgti), pirms tajā tiks nomainīti visi cipari no 0 līdz 0.



Attēlā redzama šāda pārbaude. Aktivizētais (tikko ieslēgtais) segments ir segments, kas iedegas uz pašreizējā cipara, bet nedega uz tuvākā cipara pa kreisi. Katra segmenta aktivizācija ir atzīmēta ar šī segmenta burtu. Aktivizāciju skaits ir burtu skaits visā attēlā. Tabulā ir apkopota šī pārbaude. Centrālajā kolonnā ir parādīti cipari, uz kuriem segments tika aktivizēts. Labajā kolonnā ir uzskaitīti šie cipari.

Segments	Cipars, kas izraisa segmenta aktivizēšanu	Aktivizēšanas reižu skaits
A	2, 5	2
B	7	1
C	3	1
D	2, 5, 8	3

E	2, 6, 8, 0	4
F	4, 8	2
G	2, 8	2

Visvairāk mainītais segments ir E uz ceturta cipara, un tā kods ir 4E.

Šī ir informātika

Šajā uzdevumā laika attēlošanai tiek izmantots septiņu segmentu displejs. Septiņu segmentu displejs ir izplatīts tehnoloģiskajos produktos. Tas attēlo skaitļus no 0 līdz 9, izmantojot septiņas gaismas diodes. Turklāt šis uzdevums ietver skaitīšanas sistēmas koncepciju. Skaitīšanas sistēmām ir liela nozīme datorzinātnēs. Visbiežāk izmantotā skaitīšanas sistēma ir decimālsistēma, savukārt datori datu attēlošanai izmanto bināro sistēmu. Tomēr mūsu ikdienas dzīvē pastāv arī citas skaitīšanas sistēmas. Piemēram, standarta pulkstenī minūtes tiek attēlotas sešdesmitnieku (ar bāzi 60) sistēmā, bet stundas - divdesmitčetrinieku (ar bāzi 24) sistēmā.

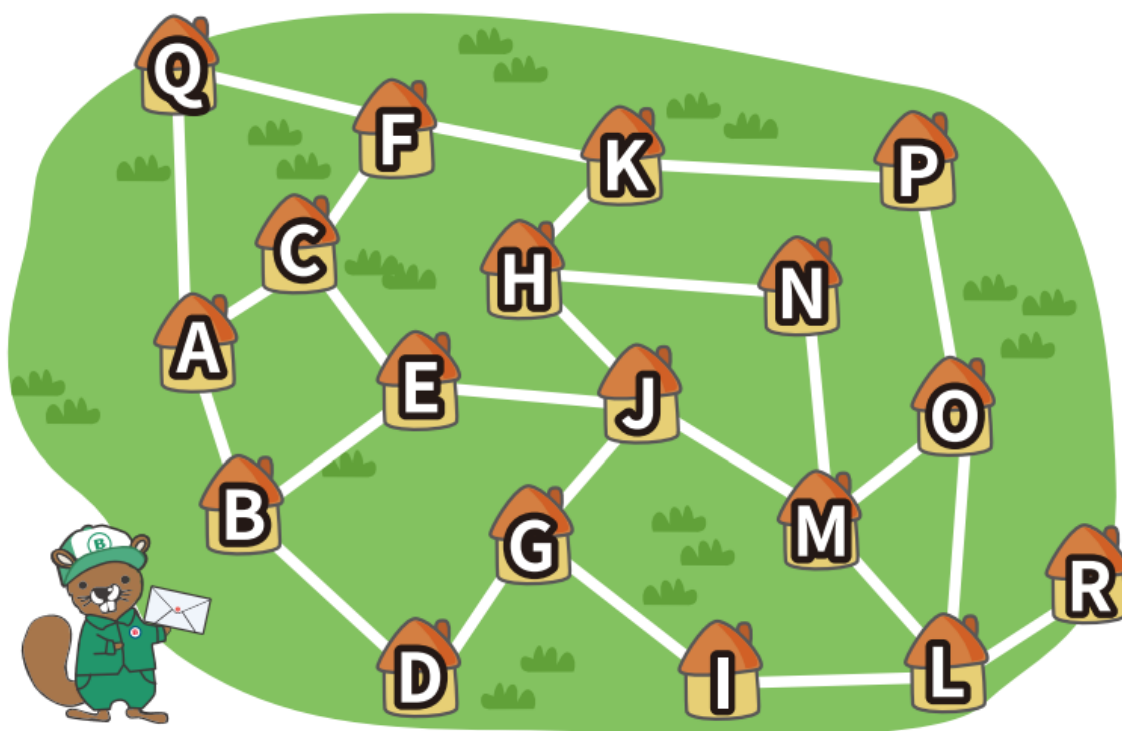
Šī ir skaitļošanas domāšana

Šis uzdevums pagē, lai skolēni izmantotu loģisko spriešanu un veselo saprātu, lai ieraudzītu, ka standarta digitālā pulksteņa pēdējais cipars mainās visbiežāk. Tāpēc uzdevumu var sašaurināt līdz segmenta izmaiņu analīzei labajā malā esošajā ciparā. Pēc tam skolēniem jāanalizē izmaiņu skaits katrā septiņu segmentu displeja segmentā pilna izmaiņu cikla laikā. Šādas izmaiņas tiek attēlotas kā displeja segmentu pāreja no izslēgta stāvokļa uz ieslēgtu stāvokli. Šim procesam ir nepieciešama spēja atpazīt modeļus.

Bebrusalas biļetens

Taiwāna

Bebrusalā ir 18 ciemati, kā parādīts attēlā zemāk. Katrā ciematā ir daudz pastnieku - ikreiz, kad no ciemata ir jānosūta ziņojums vai tajā tiek saņemts jauns ziņojums, ciemata pastnieki nākamajā dienā nogādās ziņojumu visiem saistītajiem kaimiņu ciematiem.



Piemēram, ja no ciemata A nosūta ziņojumu, ziņojuma nonākšanai ciematos B, C un Q nepieciešama 1 diena. Lai ziņojums sasniegtu ciematus D, E un F, nepieciešamas 2 dienas, un tā tālāk, līdz visi ciemati ir saņēmuši ziņojumu.

Jautājums / Izaicinājums

Ja jauns ziņojums tiek sūtīts no ciemata J, cik dienas nepieciešamas, lai tas sasniegtu visus ciematus?

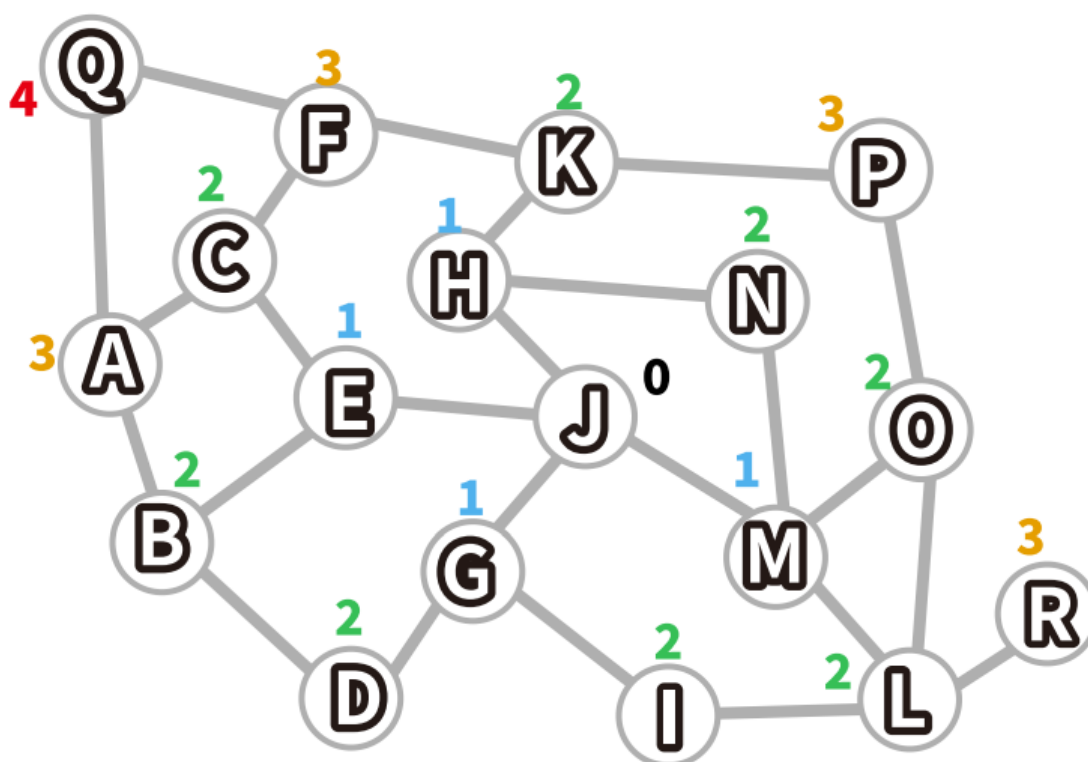
Atbilžu varianti / Interaktivitātes apraksts

Vesels skaitlis no [1, 18]

Atbildes skaidrojums

Sākot no ciemata J, mēs varam izsekot, cik dienu nepieciešams, lai ziņojums sasniegtu katru ciematu, un uzzināt atbildi:

1. diena: Ciemati E, G, H un M saņem ziņojumu.
2. diena: Ciemati B, C, D, I, K, N, O un L saņem ziņojumu, proti: ciemats B un C to saņem no E. Ciemati D un I to saņem no G. Ciemati K un N to saņem no H. Ciemati N, O un L to saņem no M.
3. diena: Līdzīgi ziņojumu saņem ciemati A, F, P un R. Šajā brīdī ziņojumu vēl nav saņēmis tikai ciemats Q.
4. diena: Ciemats Q saņem ziņojumu no F un A.



Tāpēc kopumā būs nepieciešamas četras dienas, lai nodrošinātu, ka visi ciemati ir saņēmuši ziņojumu no ciemata J.

Tā ir informātika

Šajā uzdevumā ziņojumu izplatīšanas metode ir līdzīga meklēšanas platumā (platkursija, BFS) algoritma veikšanai grafā, sistemātiski izpētot vai meklējot mezglus. Platkursijas algoritms sāk no konkrēta mezgla, vispirms apmeklējot tā blakus esošos mezglus, un pēc tam turpina apmeklēt vēl neapmeklētus blakus esošos mezglus tādā secībā, kādā tie tiek atrasti. Platkursijas pamatkonceptija ir "paplašināšana slāni pa slānim", nodrošinot,

ka vispirms tiek izpētīti mezgli, kas atrodas tuvāk sākumam, pirms paplašināšanas uz attālākiem mezgliem.

Platkursijas praktiskos pielietojumos ietilpst meklētājprogrammas, kas pārmeklē tīmekļa lapas slāni pa slānim, izmantojot saites, sociālo mediju platformas, kas analizē, kā ziņojumi izplatās no viena lietotāja citiem, un sabiedrības veselības iestādes, kas izseko kontaktus un simulē infekcijas slimību izplatību. Saistīts jēdziens ir pludināšanas-aizpildīšanas algoritms, ko parasti izmanto, lai risinātu problēmas, kas saistītas ar savienotu apgabalu paplašināšanu vai izpēti. Piemēram, pieskaroties apgabalam krāsošanas lietotnē, algoritms aizpilda visu sadaļu ar krāsu. Līdzīgi tas tiek pielietots, kad tīrīšanas robots kartē un tīra visas savienotās telpas, sākot no sākotnējās atrašanās vietas. Iekšēji plūdu aizpildīšana parasti balstās uz meklēšanu platumā vai dziļumā (dziļkursija, DFS), lai sistemātiski un rūpīgi izpētītu visus blakus esošos savienotos mežglus.

Šī ir skaitļošanas domāšana

Lai atrisinātu šo uzdevumu, studentiem vispirms ir jāsaprot un jāpielieto noteikums par to, kā ziņojumi tiek nodoti no ciemata uz ciematu. Viņiem sistemātiski jāseko ziņojuma izplatībai dienu no dienas, kas ietver algoritmisku domāšanu. Šī procesa laikā studenti iesaistās arī simulācijā, jo viņi garīgi vai vizuāli modelē, kā ziņojums laika gaitā "plūst" tīklā. Viņiem ir jāatbrīvojas no nevajadzīgām detaļām un jākoncentrējas uz galvenajiem elementiem: savienojumiem starp ciematiem un ziņojuma izplatīšanas laiku.

Informātikas atslēgvārdi un tīmekļa vietnes

Meklēšana plašumā, pludināšanas-aizpildīšanas algoritms

Būla figūras

Īrija

Būla darbības tiek izmantotas datorgrafikā. Šīs darbības ļauj no vienkāršākām figūrām veidot sarežģītākas. Zemāk ir sniegts trīs Būla pamatdarbību piemērs: UN (angliski AND), VAI (OR) un NE (NOT).

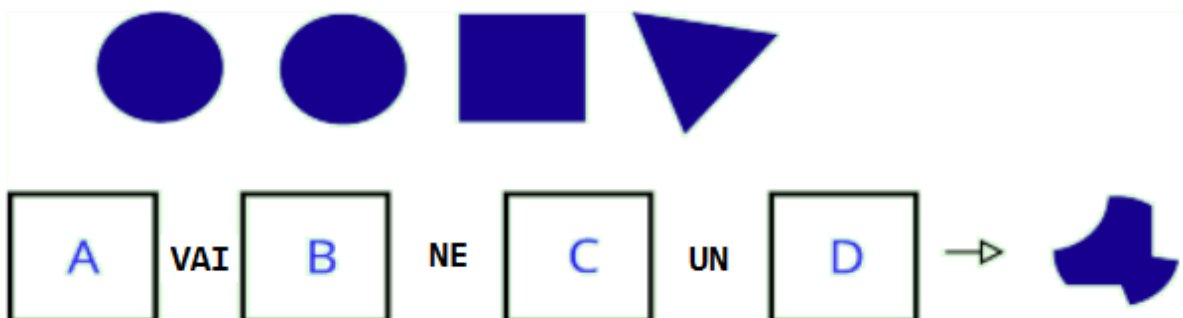
1.figūra	2.figūra	Pozīcija	Darbība		
			UN	VAI	NE
					
					

Mēs varam izmantot šīs darbības secīgi, lai uzzīmētu/izveidotu sarežģītākas figūras.

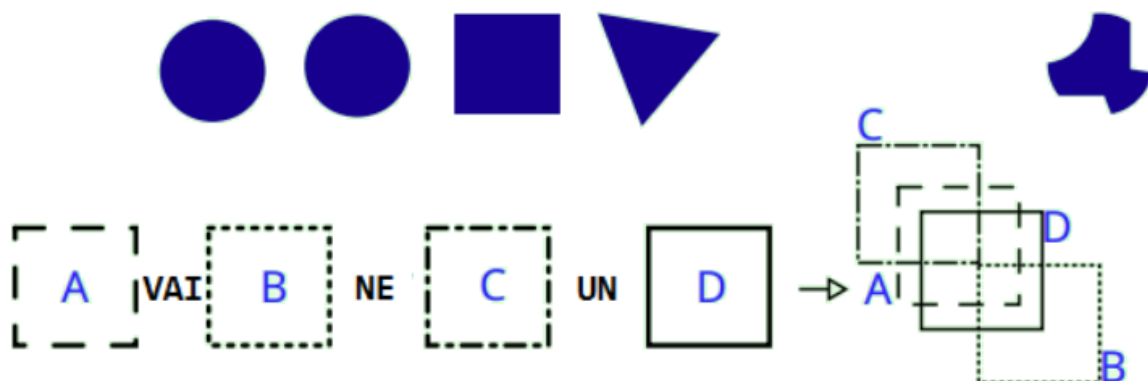


Jautājums / Izaicinājums

Jums ir dotas četras figūras un trīs Būla operācijas. Velciet un nometiet figūras pareizajās vietās tā, lai rezultāts pēc darbību izpildes būtu tāds, kāds parādīts aiz bultiņas.



Versija: Ja vēlaties sniegt skolēniem lielāku atbalstu pozīcijas noteikšanā, varat izmantot mērķa kvadrātus "pozīcija":

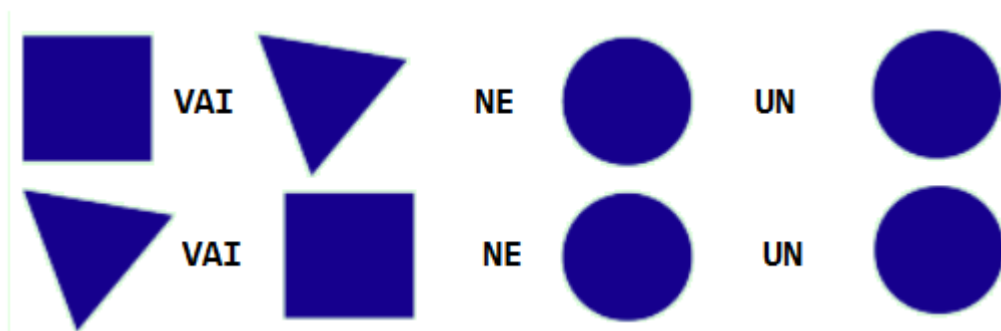


Atbilžu varianti / Interaktivitātes apraksts

Interaktivitātes apraksts: četras figūras jāvelk uz kvadrātiem, kas apzīmēti ar A, B, C, D. Ja tās atrodas tuvu atzīmētajiem kvadrātiem, tām vajadzētu "ielēkt" iekšā. Figūras var noņemt, atgriežot sākotnējā vietā (ja tās tiek pārvietotas ārpus atzīmētajiem kvadrātiem, tās atgriežas sākotnējā vietā).

Atbildes skaidrojums

Pareizā atbilde ir viena no šīm:



Lai atrisinātu šo uzdevumu, mums ir jāidentificē iegūtā attēla galvenās iezīmes.

Pirmā dotā darbība ir VAI, kas izveidos iegūtā attēla "ķermeni", apvienojot divas figūras. Šeit jums jāizmanto kvadrāts un trīsstūris. Tā kā mēs izmantojam VAI, figūru secībai nav nozīmes. VAI rezultāts ir vienāds abos gadījumos.



Nākamā dotā darbība ir NE. Tas nozīmē, ka mēs noņemsim daļu no attēla. Mēs redzam, ka mums būs jāatņem kaut kas no mūsu figūras augšējā kreisā stūra. Tā kā tas ir aplveida, mums jāizmanto aplis, lai noņemtu daļu ar NE.



Pēdējā dotā operācija ir UN, kas saglabās visu, kas ir kopīgs abās figūrās. Rezultāta iegūšanai varam izmantot pēdējo apli.



Šī ir informātika

Būla darbības jau sen tiek izmantotas datorgrafikā. Tādas darbības kā VAI (divu figūru apvienojums), UN (divu figūru šķēlums) un NE (vienas figūras "atņemšana" no citas) ļauj no vienkāršākām figūrām izveidot sarežģītākas figūras. Šīs darbības parasti ir pieejamas gan bitkartes, gan vektoru zīmēšanas programmās. Šādas darbības, veikspējas apsvērumu dēļ, labāk veikt ar vektoru attēlojumiem nekā ar bitkartēm. Tas, savukārt, ir novedis pie veselas algoritmu klases, slaucīšanas līnijas algoritmu, izmantošanas, lai efektīvi veiktu šīs darbības ar vektoru attēlojumiem. (IESPĒJA: Ja vēlaties sniegt skolēniem/skolotājiem vairāk infigūrācijas par terminu Būla loģika, tad tas attiecas uz ko tādu, kam var būt tikai viena no divām vērtībām: patiesa vai nepatiesa. Būla loģikai ir svarīga loma programmēšanā. Programmēšanā Būla apgalvojumi, piemēram, tiek izmantoti, lai izveidotu nosacījumus cikliem un zarošanās operatoriem: if punkti > 100: print("Apsveicu") Būla apgalvojums vienmēr ir patiens vai nepatiess: piemēram, "punkti > 100", "Bobs ir vecāks par Alisi", "Līst lietus". Būla operatorus, piemēram, UN, VAI, NE, var izmantot, lai izveidotu sarežģītākus apgalvojumus vai izteiksmes: piemēram, "Līst lietus" UN "Māja ir dzeltena". Šis apgalvojums ir patiens tikai tad, ja gan līst, gan māja ir dzeltena.

“Līst lietus” VAI “Māja ir dzeltena”. Šis apgalvojums ir patiess, ja līst lietus vai/un māja ir dzeltena.

NE “Māja ir dzeltena”. Šis apgalvojums ir patiess, ja māja NAV dzeltena.)

Šī ir skaitļošanas domāšana

Abstrakcija: Lai atrisinātu šo uzdevumu, skolēnam ir jāaplūko rezultātā iegūstamā figūra un jāatrod līdzības un atšķirības, lai izlemtu, kuras figūras izmantot un kādā secībā.

Sadalīšana: Mums ir jāsadala uzdevums mazākos, nosakot figūras, kas jāizmanto katrā atsevišķā darbībā.

Puķu stādīšana

Slovākija

Robots puķu dobē rindā stāda pilnībā izaugušas puķes, pamatojoties uz norādēm. Tukšā vietā nav ne norādes, ne puķes. Robots puķu stādīšanai izmanto šādas instrukcijas:

0: Doties uz vietu, kas atzīmēta ar X.

Atkārtot 1.–5. darbību, līdz vairs nav nevienas vietas ar norādi.

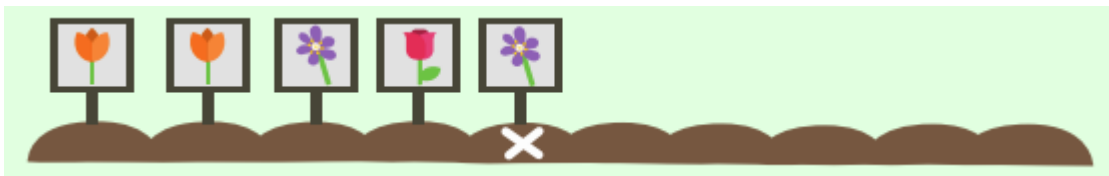
1: Ja šajā vietā ir norāde, iestādīt šajā vietā uz norādes attēloto puķi.

2: Iegaumēt tikko iestādīto puķi.

3: Noņemt norādi.

4: Dodieties pa labi, līdz sasniegta tukša vieta, pēc tam iestādīt tur puķi, ko iegaumējāt kā pēdējo.

5: Dodieties pa kreisi, līdz sasniegta vieta ar norādi vai līdz izkāpts ārpus puķu dobes.



Jautājums / Izaicinājums

Kā puķu dobe izskatīsies pēc stādīšanas?

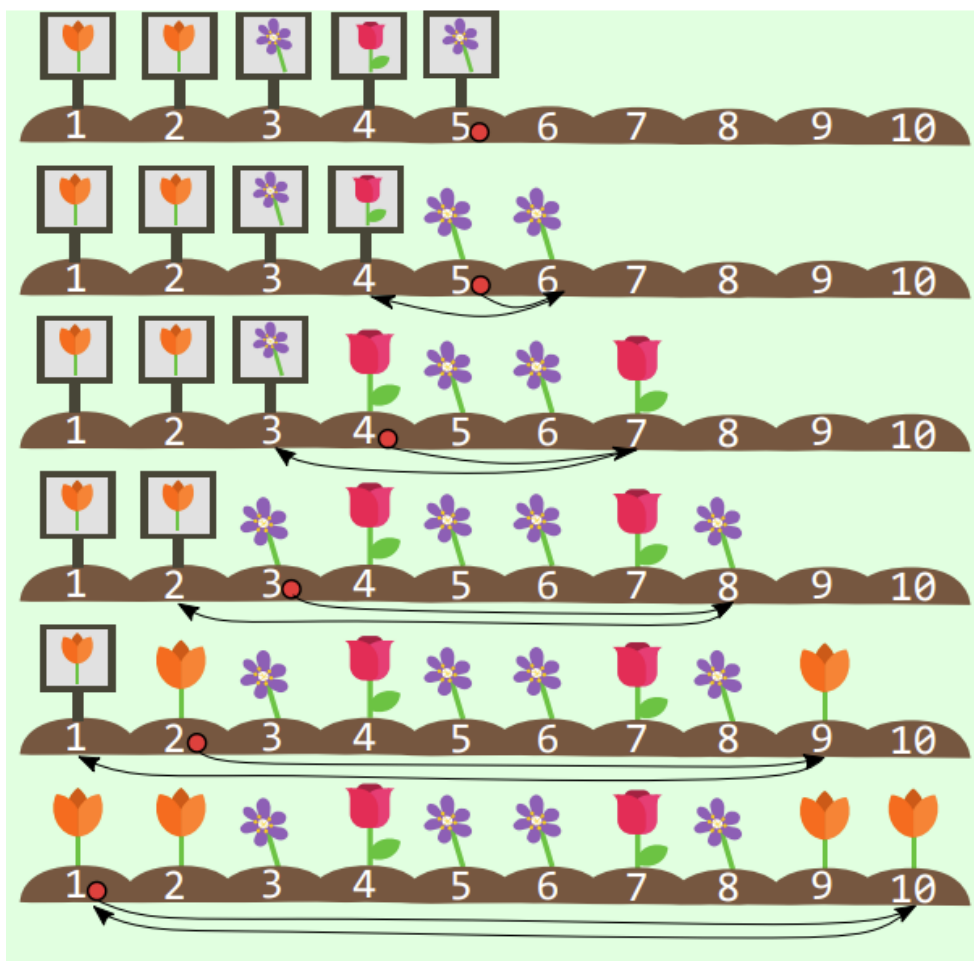
Atbilžu varianti / Interaktivitātes apraksts



Atbildes skaidrojums

Pareizā atbilde ir A.

Sanumurēsim puķu vietas. Robots sāk no pēdējās apzīmētās vietas, kas ir 5. vieta. Tur tas iestāda vijolīti, iegaumē to un iet pa labi. Tieši blakus ir tukša vieta (6. vieta), tāpēc robots iestāda pēdējo iegaumēto puķi (vijolīti) un tad iet pa kreisi, līdz atrod pirmo vietu ar norādi. Tā ir 4. vieta. Tur tas iestāda rozi un iegaumē to. Tad robots dodas pa labi uz pirmo brīvo vietu, kas ir 7. vieta, un iestāda rozi. Tālāk tas dodas pa kreisi, līdz atrod vietu ar norādi – tā ir 3. vieta. Robots tur iestāda vijolīti, iegaumē to un iet pa labi uz pirmo brīvo vietu (8. vieta), kur iestāda vēl vienu vijolīti. Gan 2. un 9. vietā, gan 1. un 10. vietā tas iestādīs tulpes. Var redzēt, ka robots tukšajās vietās stāda puķes pretējā secībā. Attēlā sarkanais punkts apzīmē robota pozīciju, un bultiņas parāda, kā tas pārvietojās.



Šī ir informātika

Šajā uzdevumā Tjūringa mašīnas koncepcija ir paslēpta aiz robota instrukcijām. Tjūringa mašīna ir mašīnas jēdziens, kas izpilda algoritmu un šajā uzdevumā tai ir viena galviņa (robots), kas pārvietojas pa lenti (puķu dobi) ar šūnām (punktiem) un var lasīt (norādes) un rakstīt (noņemt norādi un iestādīt puķes) šajos punktos. Tjūringa mašīnas atmiņa ir lente, un mašīnai var būt vairāk nekā viena lente. Šajā uzdevumā robots iegaumē, kura puķe tika iestādīta kā pēdējā, lai šo informāciju varētu ierakstīt citā lentē.

Šī ir skaitļošanas domāšana

Šajā uzdevumā ir nepieciešama algoritmiskā domāšana. Jums ir jāsaprot un jāseko piedāvātajam algoritmam, kas ir uzrakstīts kā instrukciju kopums, un jums ir jāizpilda šīs instrukcijas noteiktiem ievaddatiem. Šajā uzdevumā ievaddati ir norāžu virkne, un rezultāts (izvaddati) ir iestādīto puķu virkne.

Milti

Taivāna

Divi bebri - Alberts un Mario veido leģendāru miltu pārvadāšanas komandu. Alberts katrā braucienā pārvadā 13 kg miltu, un ceļš no maiznīcas līdz dzirnavām un atpakaļ aizņem vienu stundu. Mario pārvadā tikai 5 kg, bet katru turp un atpakaļ braucienu veic 30 minūtēs. Viņi abi nevar doties braucienā vienlaikus, jo vismaz vienam jāpaliek maiznīcā apkalpot klientus.

Tāpat kā visiem čakliem strādniekiem, viņiem kaut kad ir arī jāatpūšas. Pēc trim secīgiem braucieniem katram bebram jāpavada vismaz 30 minūtes, pirms viņš dodas nākamajā braucienā. Atpūtas laikā viņi var apkalpot klientus.



Jautājums / Izaicinājums

Alberts un Mario vēlas pārvest pēc iespējas vairāk miltu astoņu stundu laikā. Kurš no dotajiem apgalvojumiem ir pareizs?

Atbilžu varianti / Interaktivitātes apraksts

- A. Albertam jādodas braucienā pirmajam.
- B. Mario jādodas braucienā pirmajam.
- C. Mario jādodas pēdējā no braucieniem.
- D. Alberts nedrīkst būt pēdējā no braucieniem.
- E. Mario jādodas tikai vienā braucienā.

Atbildes skaidrojums

Pareizā atbilde ir A.

Galvenie fakti:

- Alberts: 13 kg vienā reizē, 1 stunda vienā reizē (13 kg/stundā)
- Mario: 5 kg vienā reizē, 30 minūtes vienā reizē (10 kg/stundā)
- Vienam bebram jābūt maiznīcā
- Katrs bebrs var izpildīt līdz trim braucieniem pēc kārtas
- Kamēr viens bebrs atpūšas maiznīcā, otrs turpina doties braucienos

Tā kā Alberts var piegādāt 13 kg/stundā, salīdzinot ar Mario 10 kg/stundā, Alberts ir efektīvāks. Tomēr mums ir jāizmanto Mario, kad Albertam ir jāatpūšas. Ja bebrs Alberts (A) pārvadā 3 reizes un Mario (M) vienu reizi, mēs iegūstam maksimālo miltu daudzumu 3,5 stundu laikā:



Tagad Alberts ir atpūties, tāpēc mēs varam atkārtot šo ciklu, līdz laiks ir beidzies:



Ja Mario sāk pirmais, tad pēc diviem (M, A, A, A) cikliem paliek tikai viena stunda. Tā kā Albertam tajā laikā ir jāatpūšas, Mario ir jāveic pēdējās divas kārtas. Tas noved pie mazāk efektīva grafika salīdzinājumā ar iepriekšējo.



Alberts nevar braukt 8 reizes, jo tas prasa 8 stundas braucienos + 2 * 30 minūšu atpūtu = 9 stundas, kas ir vairāk nekā atļauts uzdevumā. Tātad risinājums, kurā Alberts brauc 7 reizes + Mario 2 reizes, ir optimāls.

Šī ir informātika

Bebru miltu transportēšanas problēma ir plānošanas un optimizācijas problēma, jo:

- Mums ir jāplāno braucieni, ievērojot atpūtas un aizņemtības ierobežojumus.
- Mēs vēlamies maksimāli palielināt piegādāto miltu kopējo daudzumu (optimizācijas mērķis).
- Mums ir efektīvi jāsadala laiks starp Albertu un Mario.

Plānošanas un optimizācijas problēmas rodas daudzās informātikas uzdevumos, tostarp instrukciju izpildē datoru procesoros, kur aritmētiskās loģikas vienības var izpildīt tikai vienu instrukciju vienlaikus (aizņemtības ierobežojumi), instrukcijām, kas lasa datus no atmiņas, ir jāgaida, kamēr dati pienāk (līdzīgi kā atpūtai), un optimāls grafiks var izpildīt programmu daudz ātrāk nekā tad, ja grafiks ir neoptimāls.

Šī ir skaitļošanas domāšana

Šī uzdevuma risināšanā ir ieguvumi no sarežģītas problēmas sadalīšanas apakšproblēmās (dekompozīcija), atziņas, ka apakšproblēmas risinājumus var atkārtot (atkārtošana un modeļu atpazīšana), apakšproblēmas risinājumu sakārtošanas, lai atrastu optimālu risinājumu, un abstrakcijas, kas attēlo optimālo transportējamo kg/stundā. Ir nepieciešama arī loģiska domāšana, lai izslēgtu tos variantus, kas neatbilst tādiem ierobežojumiem kā atpūtas periodi un noslogojums (vismaz viens bebrs maizes ceptuvē).

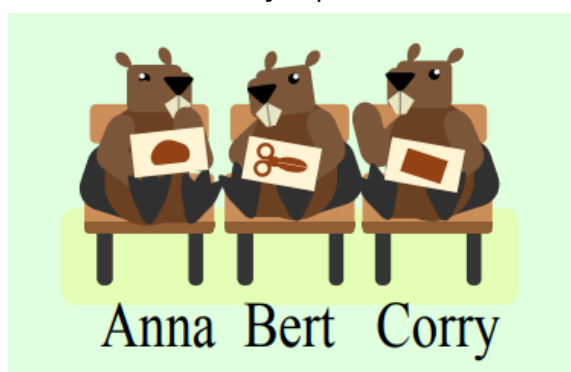
Akmens, šķēres, papīrīt's un maiņas

Brazīlija

Anna (A), Berts (B) un Korija (C) spēlē populārās spēles "Akmens-šķēres-papīrīt's" variantu. Atcerieties:

- akmens uzvar šķēres
- šķēres uzvar papīru
- papīrs uzvar akmeni

Anna, Berts un Korija apsēžas uz krēsliem un tur kārtis, lai visi tās varētu redzēt.



Tad spēlētājiem ir jāizlemj, cik kāršu maiņas viņi veiks. Maiņa ir kāršu apmaiņa starp diviem spēlētājiem. Pēc tam viņiem jāizlemj, kuri spēlētāji piedalīsies katrā maiņā.

Jautājums / Izaicinājums

Berta vienīgais mērķis ir pārspēt Koriju. Kāda stratēģija garantētu viņa panākumus?

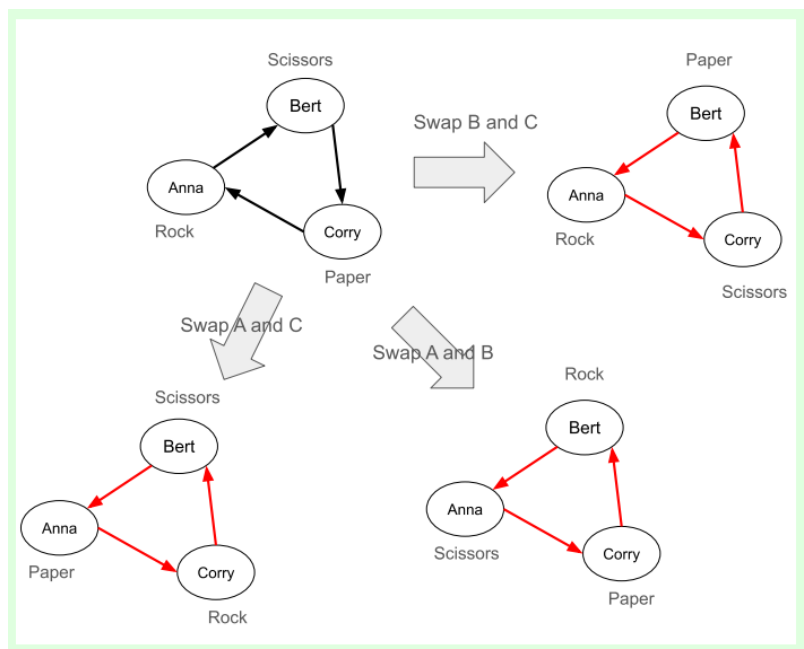
Atbilžu varianti / Interaktivitātes apraksts

- A) Bertam jāgarantē tikai nepāra skaits maiņu ar Koriju.
- B) Neatkarīgi no tā, cik maiņas viņi nolemj veikt, Bertam nekad nevajadzētu apmainīties ar kārtīm ar Koriju.
- C) Neatkarīgi no tā, cik maiņas viņi nolemj veikt, Bertam vienmēr jāapmainās ar kārtīm ar Koriju.
- D) Bertam jāgarantē tikai tas, ka maiņu skaits ir pāra skaits neatkarīgi no tā, kuras maiņas tiek veiktas.

Atbildes skaidrojums

Pareizā atbilde ir D.

Lai to atrisinātu, jāņem vērā, ka, veicot apmaiņu, neatkarīgi no tā, kura kārts ir katram spēlētājam, visas uzvaras un zaudējumi tiek apgriezti pretēji:



Šajā attēlā bultiņas norāda, kāds ir spēles rezultāts (kurš uzvar). Tāpēc pēc pāra skaita apmaiņu uzvaras ir tādas pašas kā sākotnējā situācijā. Un otrādi, pēc nepāra skaita apmaiņu uzvaras būs pretējas. Tā kā Bert vēlas uzvarēt Koriju, un tas notiek sākotnējā situācijā, viņam ir jāgarantē tikai pāra skaits maiņu.

Šī ir informātika

Šis uzdevums ir saistīts ar invariantiem, kas nozīmē atrast lietas, kas nemainās noteiktos apstākļos, vai identificēt apstākļus, kuros lietas ir tādas pašas kā iepriekš.

Invarianti ir būtiski matemātikā un datorzinātnēs, jo, ja mēs spējam tos atrast, mēs varam samazināt analizējamo gadījumu skaitu līdz mazākam skaitam, padarot uzdevumu vieglāk risināmu.

Šī ir skaitļošanas domāšana

Šablonu atpazīšana ir galvenā prasme šī uzdevuma risināšanā.

Sākumā uzdevums var šķist ļoti sarežģīts, jo šķiet, ka ir pārāk daudz iespējamo iznākumu, lai tos visus analizētu. Tomēr, ja sistemātiski analizē dažus gadījumus, nav grūti pamanīt, ka to nav tik daudz un ka tos var grupēt pēc maiņu skaita paritātes

Printera tīrīšana

Kongo demokrātiskā republika

Lai saglabātu kvalitāti, tintes printeriem pēc katra izdrukas cikla ir nepieciešams tīrīšanas cikls:

- Tīrīšanas cikls nejauci ilgst 1 vai 2 minūtes.
- Katrs izdrukas cikls var ilgt 1 vai 2 minūtes.

Frenkam ir piekļuve tintes printerim tikai 6 minūtes.

- Viņš var izlemt, vai sākt ar izdrukas vai tīrīšanas ciklu.
- Viņš var izlemt drukāt 1 vai 2 minūtes katrā izdrukas ciklā.
- Viņš nevar kontrolēt vai iepriekš zināt tīrīšanas ciklu ilgumu.
- Viņš vēlas maksimāli palielināt drukāšanas laiku.
- Viņam darbs jāpabeidz ar tīrīšanas ciklu. Pēdējam tīrīšanas ciklam ir jāsākas un jādarbojas vismaz 1 minūti, bet nav noteikti jābeidzas viņam atvēlēto 6 minūšu laikā.

Derīga grafika piemērs:

1. Frenks sāk ar tīrīšanas ciklu (kopējais laiks: 2 minūtes).
2. Frenks drukā 1 minūti (kopējais laiks: 3 minūtes).
3. Tīrīšanas cikls ilgst 1 minūti (kopējais laiks: 4 minūtes).
4. Frenks drukā 1 minūti (kopējais laiks: 5 minūtes).
5. Sākas tīrīšanas cikls.

Neatkarīgi no tīrīšanas cikla ilguma, Frenks šajā ciklā pabeidz savu 6 minūšu darbu.

Jautājums / Izaicinājums

Kura ir pirmā no 6 minūtēm, kad Frenkam ir jāsāk pirmais tīrīšanas cikls, lai garantētu, ka viņš pabeigs savu darbu ar tīrīšanas kārtu?

Atbilžu varianti / Interaktivitātes apraksts

- A. no paša sākuma
- B. pēc 1 minūtes
- C. pēc 2 minūtēm
- D. pēc 3 minūtēm

Atbildes skaidrojums

Pareizā atbilde ir: C

Viena no šī uzdevuma risināšanas stratēģijām ir sākt no beigām un strādāt atpakaļgaitā:

- Tā kā Frenkam jāpabeidz savs darbs ar tīrīšanas kārtu, tam beigās jārezervē vismaz 1 minūte. Un tā kā Frenks nevēlas, lai pēc šīs pēdējās tīrīšanas kārtas sāktos drukāšanas darbs, Frenkam beigās jārezervē tieši 1 minūte. Tāpēc Frenkam pēdējā tīrīšanas kārtā jāsāk 5 minūtes pēc sākuma. Tas nozīmē, ka Frenkam pirms pēdējās tīrīšanas kārtas būs 5 minūtes.
- Viņš var izlemt, ka, ja iepriekšējā tīrīšanas kārtā bija 1 minūti gara, viņš nosūtīs 2 minūšu drukāšanas uzdevumu, un pretējā gadījumā, ja iepriekšējā tīrīšanas kārtā bija 2 minūtes gara, viņš nosūtīs 1 minūtes drukāšanas uzdevumu. Kopā drukāšanas un tīrīšanas kārtas tad aizņems tieši 3 minūtes.
- Tad paliek 2 minūtes, ko Frenks var izmantot sākumā. Lai maksimāli palielinātu drukāšanas laiku, viņš var izlemt izmantot šīs 2 minūtes drukāšanai.

Tas nozīmē, ka pirmajai tīrīšanas kārtai jāsākas ne agrāk (un patiesībā ne vēlāk) kā 2 minūtes pēc sākuma.

Šī ir informātika

Šī ir plānošanas problēma, kas darbojas ar deterministiskiem un stohastiskiem procesiem. Deterministiskie procesi ir pilnībā kontrolējami, savukārt stohastiskos procesus ietekmē nejauši notikumi, un tos nevar pilnībā paredzēt vai kontrolēt. Šādas plānošanas piemērus var atrast vides attīrīšanā, projektu vadībā, naftas izpētē, sensoru plānošanā mobilajos robotos un cikla laika modelēšanā, kā arī daudzās citās jomās.

Uzdevuma idejas ir ņemta no Nim spēles. Sērkociņu vietā (visizplatītākais modelis Nim spēlēs) mums ir tīrīšanas un drukāšanas ciklu ilgums ar diviem spēlētājiem: Frenku un "Nejaušas tīrīšanas procesu". Nim spēle ir tā, ko spēļu teorijā sauc par nulles summas spēli. Nim spēles bieži izmanto, apmācot mākslīgo intelektu, īpaši ieviešot pastiprināšanas mācīšanos, lai parādītu, kā dators var iemācīties spēlēt spēli, to spēlējot.

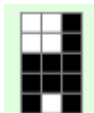
Ciparu mikslis

Čehija

Stundas laikā Džemmai ir garlaicīgi, un viņa sāk aizpildīt kvadrātus savā lapā, lai attēlotu skaitļus. Viņa izdomā īpašu veidu, kā attēlot savu iecienītāko divciparu skaitli, 42: no diviem ciparu "4" un "2" attēlojumiem 3×5 režģī, kas parādīts zemāk, viņa izveido jaunu režģi, kur katrs kvadrāts ir melns tad un tikai tad, ja tieši viens no diviem atbilstošajiem kvadrātiem attēlos "4" un "2" ir melns:

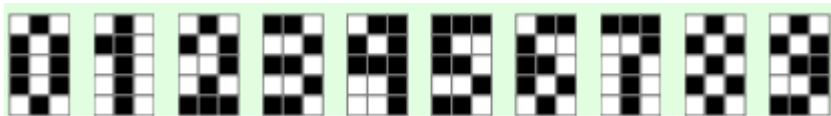


Džemmai izmantoja šo pašu metodi, lai attēlotu citu divciparu skaitli:



Jautājums / Izaicinājums

Kuru skaitli Džemmai attēloja, zinot, ka ciparu sākuma attēli ir šādi? Ja ir vairākas atbildes, norādiet tikai vienu no tām.



Atbilžu varianti / Interaktivitātes apraksts

Vesels skaitļi no 10 (ieskaitot) līdz 99 (ieskaitot)

Kā papildinājumu var piedāvāt divus interaktīvus 3×5 režģus kopā ar aprēķinātu trešo režģi ar tādu pašu izmēru, kas parāda pirmo divu režģu pikseļu XOR.

Atbildes skaidrojums

Divas pareizās atbildes ir 26 un 62. Apvienojot šos divus ciparus, iegūst vēlamu attēlu:



Nemiet vērā, ka, apvienojot tos apgrieztā secībā, joprojām iegūsiet to pašu attēlu, jo to apvienošanas metode ir simetriska attiecībā pret tiem.

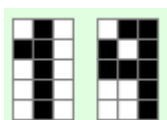
Lai atrisinātu šo uzdevumu, mums jāatrod veidi, kā noteikt divus pareizos ciparus, vienlaikus izvairoties no visu iespējamo kombināciju meklēšanas. To ir vienkārši par daudz: kopā 100, no "00" līdz "99". Tā kā viencipara skaitļos mēs parasti izlaižam pirmo 0 (mēs rakstītu "7", nevis "07"), tas samazina kombināciju skaitu līdz 90 vai pat 45, ja ņemam vērā iepriekš minēto simetriju, taču tas joprojām ir par daudz, lai to pilnībā pārbaudītu.

Pirmā lieta, kas jāsaprot, ir tā, ka apvienošanas darbība dod melnu pikseli dotajā pozīcijā ikreiz, kad šis pikselis atšķiras divos avota ciparos. Tas izriet no tā, ka rezultātā iegūstam melnu pikseli ikreiz, kad mums ir tikai viens melns pikselis, ja ņemam vērā avota attēlus. Ja aplūkojam vienu pikseli, apskatot visas iespējas, iegūstam šādu atbilstības tabulu:

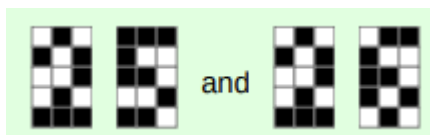
Pikselis pirmajā attēlā	Pikselis otrajā attēlā	Melno pikseļu skaits	Pikselis kombinētajā attēlā
Balts	Balts	0	Balts
Melns	Balts	1	Melns
Balts	Melns	1	Melns
Melns	Melns	2	Balts

Apzinoties to, mēs varam uzreiz izslēgt visus skaitļus, kas sastāv no viena un tā paša cipara divas reizes, piemēram, "22" vai "99", jo visi pikseļi starp abiem cipariem vienmēr būtu vienādi, un tas radītu pilnīgi baltu kombinēto attēlu.

Viens no veidiem, kā pieiet risinājuma meklēšanai, ir saprast, ka kombinētā attēla labā malējā kolonna ir pilnīgi melna, kas norāda, ka abiem avota cipariem šajā kolonnā jābūt visiem dažādiem pikseļiem. Viens acīmredzams šāds ciparu pāris ir "1"–"4" — pēdējā kolonna ir pilnīgi tukša, ja ir "1", un pilnībā pilna, ja ir "4":



Diemžēl, aplūkojot, piemēram, otro rindu vai apakšējo rindu, mēs izslēdzam šo pāri, jo tas vairs neatbilst kombinētajam attēlam. Divi citi nedaudz mazāk acīmredzami pāri, kuros visi pikseļi pēdējā kolonnā ir atšķirīgi, ir "2"–"5" un "2"–"6":



Aplūkojot atlikušos kombinētā attēla pikseļus (piemēram, pašu pirmo pikseli augšējā rindā), mēs varam izslēgt "2"–"5". Mums paliek pāris "2"–"6", kas atbilst visiem pikseļiem. Ņemot vērā iepriekš minēto simetriju, mēs secinām, ka gan 26, gan 62 ir iespējamās atbildes.

Šī ir informātika

Centrālā operācija, kas tiek izmantota kombinētā attēla izveidē, faktiski ir loģiskā operācija "izslēdzošais VAI" (ExclusiveOR, XOR). Būla operācija darbojas ar patiesām/aplamām vērtībām (sauktas par loģiskām vērtībām) un iegūst citu loģisko vērtību. Šī ir loģiskās operācijas XOR patiesuma tabula, kur Z ir izvade:

A	B	Z = XOR(A, B)
0	0	0
1	0	1
0	1	1
1	1	0

Līdzību ar iepriekš parādīto pikseļu atbilstības tabulu var pamanīt, ja 0 aizstājam ar tukšiem kvadrātiem un 1 ar melniem kvadrātiem.

Citas loģiskās operācijas ir UN un VAI. Šādas operācijas tiek plaši izmantotas aparatūras sistēmu projektēšanā loģisko vārtu veidā. Tie ir pamatelementi, kas saņem ievadā divus signālus un "aprēķina" izvadi saskaņā ar patiesuma tabulu.

Kriptogrāfijā XOR ir centrālā loma simetriskās šifrēšanas shēmās, jo tā ir atgriezeniska: divas reizes piemērojot vienu un to pašu XOR operāciju ar vienu un to pašu atslēgu, tiek atjaunota sākotnējā vērtība. (Patiesām, šeit, apvienojot apvienoto attēlu ar vienu no divu ciparu attēliem, tiks iegūts... otra ciparu attēls!) Šī īpašība padara XOR noderīgu gan datu kodēšanai, gan dekodēšanai. XOR tiek plaši izmantots arī kļūdu noteikšanas un labošanas metodēs, piemēram, kontrolsummu un cikliskās redundances pārbaudēs (CRC), kur tas palīdz identificēt neatbilstības starp pārraidītajiem un saņemtajiem datiem.

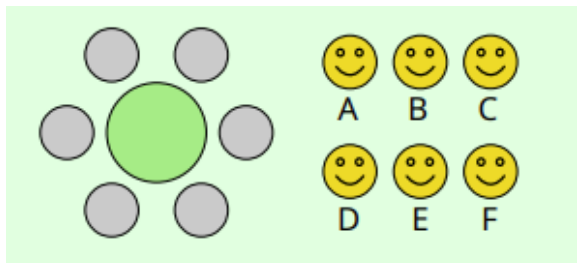
Šī ir skaitļošanas domāšana

Šis uzdevums galvenokārt apmāca paraugu saskaņošanu, pārbaudot pikseļus avota attēlos un apvienotajā attēlā.

Priecīgā aplī

Lietuva

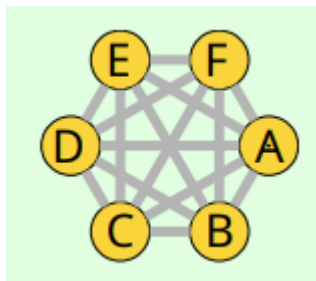
Seši draugi vēlas vakariņot pie apaļa galda. Visi no viņiem dod priekšroku sēdēt blakus saviem labākajiem draugiem un jūtas skumji, ja sēž blakus kādam, kas viņiem nepatīk.



Jautājums / Izaicinājums

Sakārtojiet draugus ap galdu, velkot un nometot viņu simbolus uz krēsliem. Ja viņi tiek apsēdināti blakus kādam, kas viņiem nepatīk, seja kļūs sarkana un skumja. Atrodiet piemērotu sēdvietu, lai visi būtu laimīgi un smaidīgi.

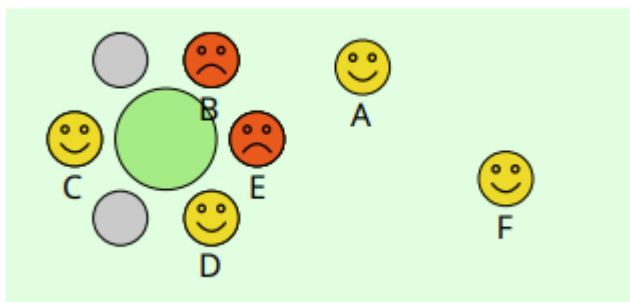
Zemāk esošā interaktīvā diagramma dota, lai palīdzētu jums atrisināt uzdevumu. Atbilde nav atkarīga no tās. Varat noklikšķināt uz saitēm, lai mainītu to krāsu.



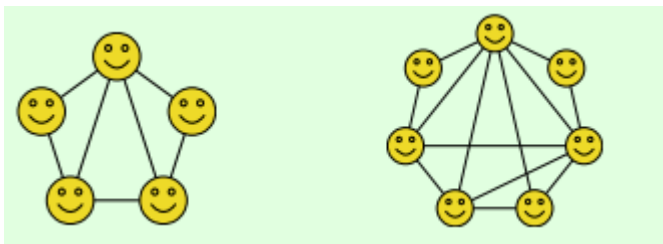
Atbilžu varianti / Interaktivitātes apraksts

Interaktīvo versiju var pārbaudīt šeit:

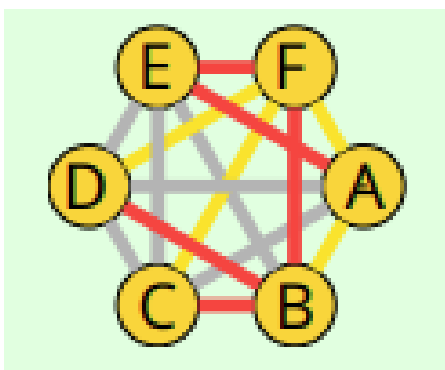
<https://bebras-lodge.sp.fsf.vu.lt/public/d8fa0bc9897a40b09b65>



Draugus (smaidīgās sejas) var pārvietot un vilkt uz krēsliem (pelēki aplī). Sākumā visi cilvēki atrodas sānos un smaida. Kad cilvēku velk uz krēsla, ja viņam nepatīk vismaz viens blakus sēdošs cilvēks, smaids mainās uz sarauktu pieri. Cilvēku, kuriem patīk viens otram, grafs tiek ģenerēts šādi: vispirms ņemam attēlā redzamo grafu un katru tur savienoto cilvēku pāri, kas viens otram patīk, un katrā nesavienotajā pārī abi cilvēki viens otru ienīst (vienkāršības labad pietiktu ar to, ka viens cilvēks ienīst otru). Tad mēs varētu katrai personai piešķirt vārdus nejauši, bet šādā gadījumā kādam varētu paveikties un nejauši pareizi novietot cilvēkus ap galdu. Tātad sākumā visi savienojumi apzīmētajā grafā ir nezināmi jeb atrodas "kvantu stāvoklī". Tiklīdz divi cilvēki tiek nomesti blakus viens otram, stāvoklis "sabrūk" – viņi vai nu sāk viens otram patīkt, vai nepatīkt. Lai izvēlētos visus iespējamās $6! = 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 720$ cilvēku permutācijas grafikā tiek pārbaudītas un vai nu atmestas (jo tās ir pretrunā ar jau zināmu informāciju), vai arī tām tiek piešķirts svars kā katra zināmā pāra vērtību summa (1 nepatīkai vienam pret otru, 2 patikai diagonālēs, 0 patikai sešstūra malās dotajā attēlā). Pēc tam tiek izvēlēta šķautne ar permutāciju ar vislielāko svaru (vai nejauši, ja svāri ir vienādi). Ņemiet vērā, ka attēlā redzamajam grafam ir maksimālais šķautņu skaits ar tieši vienā Hamiltona ciklā. Šeit ir grafi uzdevuma vieglākajai/grūtākajai versijai:



Uzdevums bez norādījumiem var būt milzīgs, tāpēc tiek dota interaktīva diagramma, lai atzīmētu zināmās saiknes: noklikšķinot, mainās dzeltenā/sarkanā/pelēkā krāsa, piemēram:



Atbildes skaidrojums

Viens no veidiem, kā atrisināt uzdevumu, ir secināt, kurai personai patīk kuri citi cilvēki, mēģinot novietot visus cilvēku pārus uz krēsliem blakus, tādējādi ģenerējot grafiku, kas ir homomorfs (t. i., ar tādu pašu struktūru) kā iepriekš redzamajā attēlā. Tad mēs varam

atrast Hamiltona ciklu (t. i., ciklu, kas katru virsotni apmeklē tieši vienu reizi) grafā un atbilstoši tam izvietot draugus ap galdu. Vēl viens veids ir meklēt Hamiltona ciklu, vienlaikus atvasinot grafu: sākumā novietojiet jebkuru personu uz jebkura krēsla, tad jebkuru citu blakus, līdz abi ir laimīgi, un tā tālāk, un, ja mēs nonākam pie situācijas, kad jebkuras personas novietošana kādu padara nelaimīgu, mēs noņemam vienu personu (atpakaļejoša izsekošana) un turpinām meklēšanu. Ar šo grafu, ja permutācija tiktu izvēlēta nejauši, būtu (iespējams) $7/60$ varbūtība ($1/30$ versijā ar 5 cilvēkiem) pareizi uzminēt secību bez atpakaļejošas izsekošanas.

Šī ir informātika

Šis uzdevums ir balstīts uz informātikas pamatjēdzieniem, kas saistīti ar grafu teoriju un algoritmisku problēmu risināšanu. Konkrēti, tas pēta Hamiltona ceļus un ciklus — maršrutus grafā, kas katru mezglu apmeklē tieši vienu reizi (ceļam) vai vienu reizi un atgriežas sākuma punktā (ciklam). Šīs problēmas ir ļoti aktuālas reālās pasaules lietojumprogrammās, piemēram, maršrutu optimizēšanā transportā un loģistikā, kur ir ļoti svarīgi samazināt braucienu skaitu, vienlaikus aptverot visus galamērķus.

No skaitļošanas viedokļa Hamiltona ceļu vai ciklu atrašana pieder pie problēmu klases, kas pazīstama kā NP-pilnas, kas nozīmē, ka tās ir ļoti grūti efektīvi atrisināt, palielinoties grafa izmēram. Nav zināms algoritms, kas varētu ātri (polinomiālā laikā) atrisināt visus gadījumus, kas padara šo par labu skaitļošanas ziņā sarežģītas problēmas piemēru.

Lai risinātu šādas problēmas, bieži tiek izmantots atpakaļejošas izsekošanas algoritms — stratēģija, kas sistemātiski izpēta visas iespējamās iespējas, atmet tās, kas ved uz strupceļu, un atkāpjas, lai izmēģinātu alternatīvas. Šī pieeja parāda, kā informātika nodarbojas ne tikai ar problēmu risināšanu, bet arī ar skaitļošanas sarežģītības novērtēšanu, efektīvu algoritmu izstrādi un izpratni par to, ko var aprēķināt saprātīgā laika periodā.

Strādājot ar Hamiltona ceļiem, studenti tiek iepazīstināti ar galvenajām informātikas jomām: datu struktūrām (grafiem), algoritmisko domāšanu (atpakaļejoša izsekošana), optimizācijas un sarežģītības teoriju — tas viss ir būtiski reālās pasaules problēmu risināšanai, izmantojot skaitļošanas metodes.

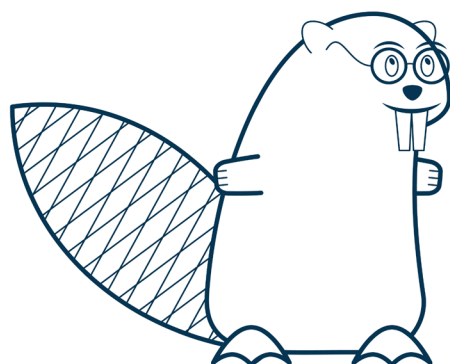
Šī ir skaitļošanas domāšana

Hamiltona ceļa un cikla problēmu risināšanā ir jāpielieto vairākas būtiskas skaitļošanas domāšanas stratēģijas. Šīs problēmas ietver katra mezgla apmeklēšanu grafā tieši vienu reizi un atgriešanos sākuma punktā (cikla gadījumā), kas atspoguļo reālās pasaules

scenārijus, piemēram, maršrutēšanu vai plānošanu. Lai to panāktu, studenti izmanto grafu šķērsošanas metodes, lai sistemātiski izpētītu savienojumus starp mezgliem. Palielinoties grafa izmēram, iespējamo ceļu skaits strauji pieaug, padarot izsmelošo meklēšanu nepraktisku. Tāpēc tiek izmantota atpakaļizsekošana, lai izpētītu ceļu, atklātu strupceļus un atgrieztos pie iepriekšējiem soļiem, lai izmēģinātu alternatīvas.

Modeļu dedukcija palīdz sašaurināt meklēšanas telpu, identificējot struktūras vai konfigurācijas, kas nekad nevar novest pie derīgiem risinājumiem. Var izmantot arī heuristikās meklēšanas stratēģijas, lai virzītu meklēšanu uz daudzsološākām grafa zonām, prioritāri piešķirot ceļiem, kas, pamatojoties uz daļējām zināšanām, visticamāk, novedīs pie panākumiem.

Kopā šīs pieejas atbalsta loģiskās spriešanas, stratēģiskās izpētes un efektīvas problēmu risināšanas attīstību, kas viss ir skaitļošanas domāšanas pamatā.



copyright Bebras