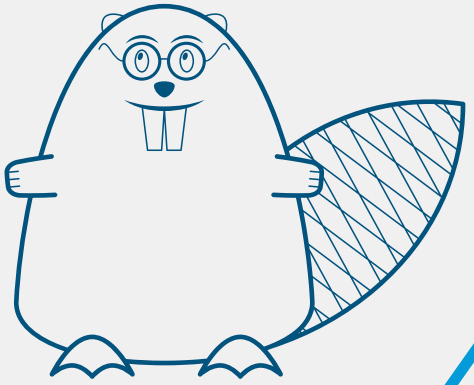
VISMA

Bebr[a]s

2019. gada
1. kārtas uzdevumi ar atbildēm
5.-6. klasei



SATURS

Failu kārtošana

Austrija

Toms, Pēteris, Anna, Marks, Dženija un Džoanna ir draugi.

Toms glabā tekstu failus ar stāstiem par saviem piedzīvojumiem ar draugiem mapē “Mūsu-stāsti”. Katrā no stāstiem ir pieminēts viens vai vairāki draugi. Viņš vēlas salikt kopā visus failus, kuros ir pieminēta “Anna”. Šim nolūkam viņš izmantoja programmu, kas pārkopē visus failus, kuros ir pieminēta “Ann” vai “ann” (ņemot vērā, ka latviešu valodā vārda galotnes locījumos var atšķirties) jaunā mapē ar nosaukumu “Annas-stāsti”.

Tomēr tad Toms saprata, ka šajā mapē varētu būt iekļuvuši arī faili, kuros ir pieminēta “Džoanna”. Viņš ieslēdz programmu otro reizi, no “Annas-stāsti” pārvietojot atpakaļ uz “Mūsu-stāsti” visus failus, kuros tika pieminēta “Džoanna”.

Pēc tam, kad Toms failu kārtošanu bija pabeidzis, četri no draugiem izteica šaubas par rezultāta pareizumu:

Pēteris: Ne visi stāsti par Annu atrodas mapē "Annas-stāsti"

Anna: Daži faili, kur minēta "Džoanna", joprojām atrodas "Annas-stāsti"

Marks: Daļa stāstu var būt pazaudēti (izdzēsti)

Džoanna: Mapē "Annas-stāsti" atrodas stāsti, kuros Anna nemaz nav pieminēta.

Uzdevums:

Kāds ir lielākais skaits no šiem izteikumiem, kas vienlaikus var būt patiesi?

Pareizā atbilde: 2

Pētera atbilde ir pareiza jeb Toms pieļāva šo kļūdu, jo stāsti, kuros Anna minēta kopā ar Džoannu tiks pārvietoti uz mapi “Mūsu-stāsti”.

Anna: Šāda atbilde nav iespējama, jo visi faili, kas satur Džoanna tiks pārceli uz “Mūsu-stāsti”.

Marks: Šāda kļūda nav iespējama, jo stāsti būs palikuši vai nu “Annas-stāsti” vai pārceli uz “Mūsu-stāsti”.

Džoanna: Tā kā sākotnēji mapē tika iekļauti stāsti, kas satur “ann” un “Ann”, tad “Annas-stāsti” atrodas tikai stāsti par Džoannu un Annu. Otro reizi iedarbinot sistēmu tiks

pārvietoti stāsti, kuros ir pieminēta tikai Džoanna vai gan Anna, gan Džoanna. Tādējādi mapē atradīsies tikai un vienīgi stāsti par Annu.

Kā tas ir saistīts ar informātiku?

Datņu meklēšana ir būtisks uzdevums. Datnes var meklēt gan pēc nosaukuma, gan pēc satura. Meklēšana var būt reģistrjūtīga (kad vēlamies atrast precīzi meklēto simbolu virkni un lielo/mazo burtu lietojums ir svarīgs), vai arī neatkarīga no reģistra (lielie un mazie burti netiek šķiroti). Risinot šķietami vienkāršus meklēšanas uzdevumus, ir viegli tajos pieļaut loģiskas kļūdas. Vienmēr ir svarīgi pārbaudīt iegūto rezultātu vai nu pārskatot meklējamo datu kopu, vai arī pielietojot alternatīvu meklēšanas metodi.

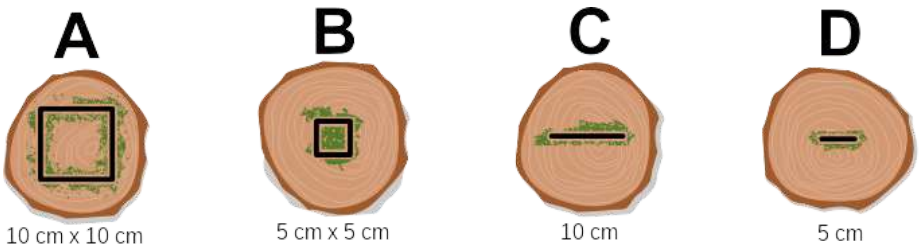
Programmēšanā vienkāršu teksta meklēšanu var veikt ar "RegEx" (https://regexr.com/) palīdzību (nosaukums atvasināts no vārdu salikuma "regulāras izteiksmes"). RegEx ir standarta veids kā atrast noteiktiem kritērijiem atbilstošu simbolu virkni dotajā tekstā.

Zīmogi

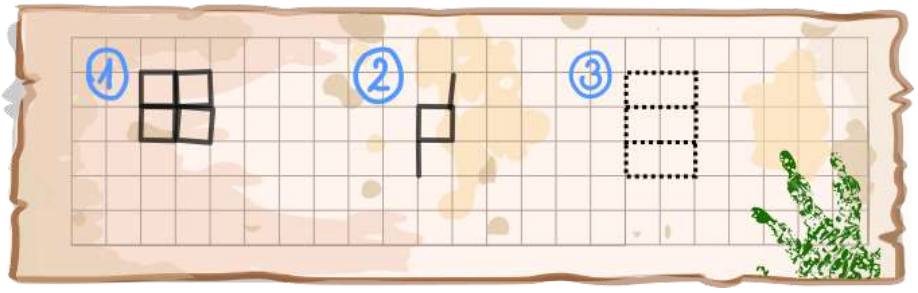
Šveice

punkti

Bebram Paulam ir 4 zīmogi: A, B, C un D:



Izmantojot tos, viņš ir izveidojis 1. un 2. figūru kā redzams attēlā zemāk:



Lai izveidotu 1. figūru, Pauls izmantoja B zīmogu četras reizes.
Lai izveidotu 2. figūru, Pauls izmantoja B zīmogu vienreiz un D zīmogu divreiz.
Tagad Pauls vēlas izveidot arī 3. figūru, un viņa draudzene Marija grib palīdzēt.

Uzdevums:

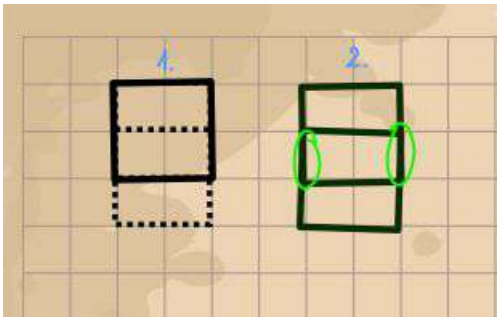
Marija apgalvo, ka 3. figūru var izveidot, izmantojot tikai vienu zīmogu divreiz. Ar kuru zīmogu tas ir iespējams?

Atbilžu varianti:

A) Lielais kvadrāts	B) Mazais kvadrāts	C) Garā līnija	D) Īsā līnija
			

Pareizā atbilde: A

Marija darbojas attapīgi, izmantojot pārklāšanās metodi ar zīmogu A  . Pirmajā solī viņa uzspiež augšējo figūras daļu, bet otrajā - apakšējo. Zaļie aplīši attēlā norāda uz vietām, kurās zīmogs ir uzspiests divreiz jeb pārklājas.



Kā tas ir saistīts ar informātiku?

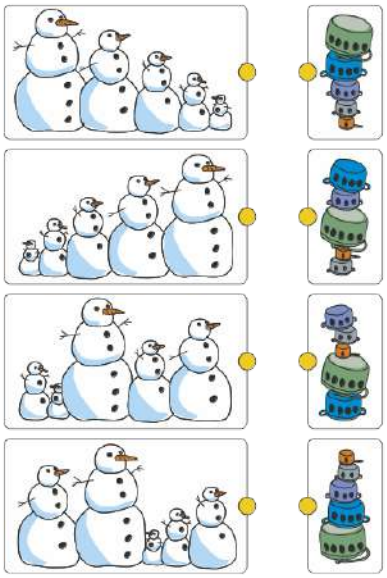
Uzdevumā doto figūru var izveidot arī daudzkārt lietojot zīmogu “Īsā līnija”. Daudziem uzdevumiem ir liels skaits arisinājumu, kas noved pie pareiza rezultāta. Bieži vien kādus atrisinājumus atrast ir vieglāk nekā citus, bet ne vienmēr visi atrisinājumi ir līdzvērtīgi.

Piemēram, var atšķirties veicamo soļu vai, datora gadījumā, izpildāmo instrukciju skaits, kas ietekmē nepieciešamo datorresursu apjomu. Šajā uzdevumā efektīvāks būs atrisinājums, kas izmantos mazāku zīmogu skaitu un izmantošanas reižu skaits būs mazākais iespējamais. Informātikas pamatbūtība arī ir atrast visefektīvāko (optimālo) atrisinājumu starp, iespējams, ļoti daudziem derīgiem.

Sniegavīru cepures

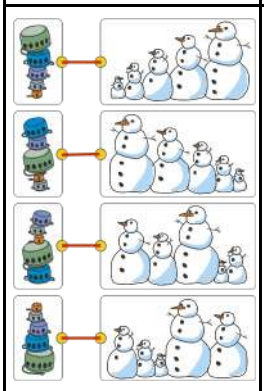
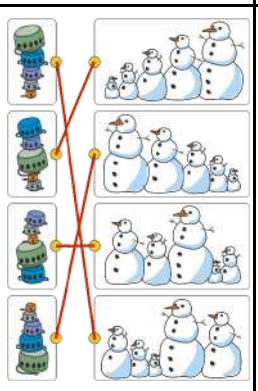
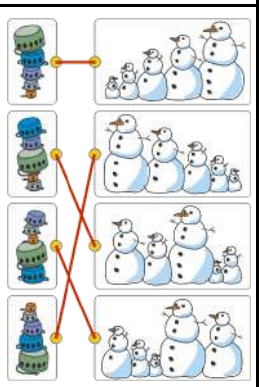
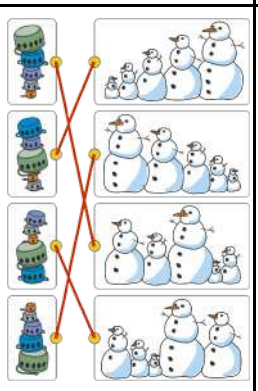
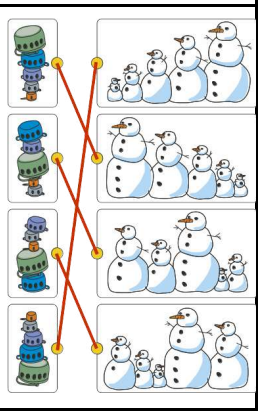
Lietuva

Pieci sniegavīri stāv rindā. No labās uz kreiso pusi katram pieder viena cepure atbilstoši izmēram. Sniegavīri cepures var dabūt ņemot no augšas pa vienai.

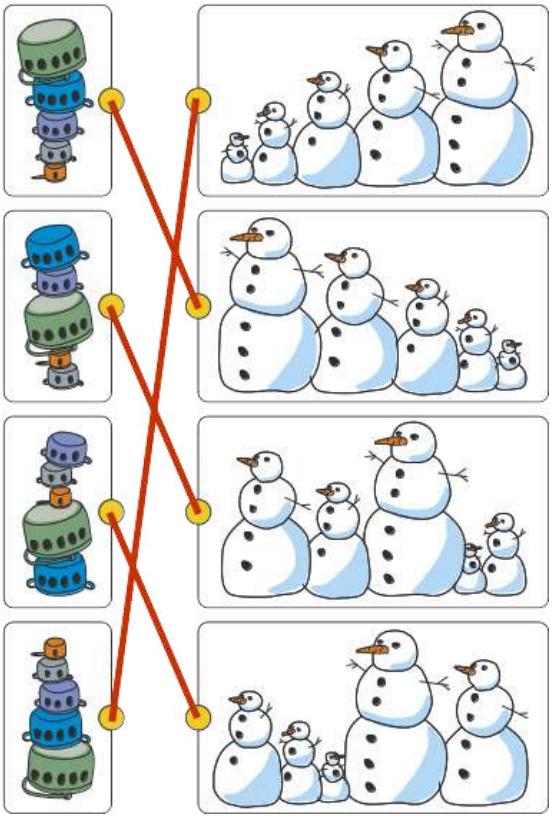


Uzdevums:

Kuras cepuru kaudzes pieder katrai sniegavīru rindai? Savieno atbilstošos attēlus.
Būs interaktīvs uzdevums, nevis atbilžu varianti.

A)	B)	C)	D)	E)
				

Pareizā atbilde: E)



Pirmā cepuru kaudze pieder otrajai sniegavīru rindai. Pirmais sniegavīrs ir vislielākais (ar 5 pogām), un pirmā cepure no augšas ir vislielākā. Otrajam sniegavīram pieder nākamā cepure utt.

Otrā cepuru kaudze pieder trešajai sniegavīru rindai. Pirmais sniegavīrs ir otrs lielākais ar 4 pogām, un pirmā cepure ir otra lielākā arī ar 4 pogām. Tāpat kā iepriekš tiek piešķirtas cepures visiem pārējiem.

Trešā cepuru kaudze pieder ceturtajai sniegavīru rindai. Šeit pirmais sniegavīrs ir trešais lielākais ar 3 pogām, un pirmā cepure ir trešā lielākā. Arī te kā iepriekš pārējās cepures tiek attiecīgi sadalītas pārējiem sniegavīriem, un tā ar katru cepuru kaudzi un sniegavīru rindu.

Atbilde A ir nepareiza, jo pirmajā rindā mazākajam sniegavīram ar vienu pogu pienāktos lielākā cepure.

Atbilde B ir nepareiza, jo pirmajā rindā mazākajam sniegavīram pienāktos otra lielākā cepure.

Atbilde C ir nepareiza, jo pirmajā rindā mazākajam sniegavīram pienāktos lielākā cepure.

Atbilde D ir nepareiza, jo pirmajā rindā mazākajam sniegavīram pienāktos otra lielākā cepure.

Kā tas ir saistīts ar informātiku?

Kad jūs katru sniegavīru saistāt ar viņa cepuri, nemainot secību ne cepuru, ne sniegavīru rindā, patiesībā tiek veikta sniegavīru un cepuru rindas elementu kartēšana. Vispirms ir jāatbilst šo rindu pirmajiem elementiem (augšējai cepurei un sniegavīram, kas atrodas visvairāk pa kreisi). Pēc tam ir jāatbilst nākamajiem pēc kārtas rindu elementiem, un tā līdz

rindu beigām kārtējam vienas rindas elementam jāatbilst kārtējam elementam no otras rindas.

Cepures tiek dotas veidā, kam datorā atbilst abstrakta datu struktūra (ADS) steks. Ar datiem šādā struktūrā var veikt tikai noteikta veida operācijas:

- Paņemt virspusē esošu elementu
- Pievienot jaunu elementu steka esošo elementu virspusē

Cita ADT ir rinda, ar kuras elementiem var veikt šādas operācijas:

- Paņemt pirmo rindas elementu
- Pievienot jaunu elementu rindas beigās.

Šajā uzdevumā dabiski ir pieņemt, ka cepuru kaudze ir veidota kā steks, liekot katru nākamo cepuri cepuru kaudzes virspusē, bet sniegavīri rindā stājušies pēc kārtas, nākamajam stājoties rindas beigās.

Meklējot pareizo atbildi, nākas veikt šo datu struktūru elementu kartēšanu.

Tīmekļa vietnes:

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

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

https://en.wikipedia.org/wiki/Convolution_(computer_science)

Stāvvieta

Vācija

Mašīnas stāvvietā var novietot brīvajās vietās vai priekšā citām mašīnām, kā tas ir redzams zemāk attēlā.

Mašīnas, kas novietotas priekšā citām, var pastumt uz priekšu vai atpakaļ, ja tās bloķē citu mašīnu, kas vēlas braukt prom.

Piemēram, attēlā mašīna A nav bloķēta, tāpēc tā var netraucēti doties prom, bet mašīnu L ir nobloķējusi mašīna M, tāpēc M ir nepieciešams pastumt, lai L varētu doties prom.

Uzdevums:

Kura mašīna nevar izbraukt, ja netiek pārvietotas divas citas mašīnas?

Pareizā atbilde: Mašīnai I

Mašīnu I bloķē mašīna N. Nav pietiekami daudz vietas, lai pārvietotu N mašīnu tik tālu, ka I var atstāt savu vietu. Tādēļ mašīna O ir jāpabīda pa kreisi vai mašīna M jāpastumj pa labi. Tikai tad pietiek vietas, lai pastumtu mašīnu N prom tik tālu, lai tā netraucētu mašīnas I izbraukšanai.

Nevienas citas mašīnas izbraukšanai nav nepieciešams pārvietot divas citas mašīnas.

A, D, E, J un Q var atstāt savas stāvvietas uzreiz. Mašīnas B, C, F, G, H, K un L var izbraukt no stāvvietas, pārvietojot P, O, N vai M.

Kā tas ir saistīts ar informātiku?

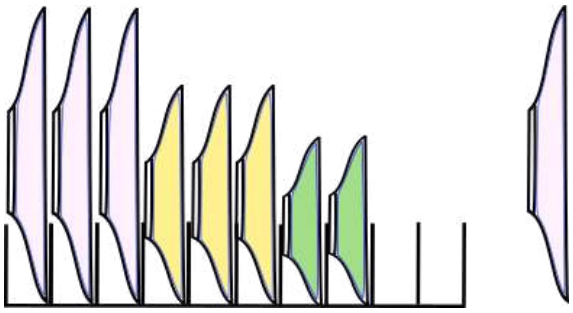
Šajā uzdevumā tiek izmantoti divi būtiski ar informātiku saistīti aspekti:

1) Izsmeļošās meklēšanas (pilnās pārlases) algoritms, kas visām mašīnām-kandidātēm pārbauda, kurai no tām piemīt vajadzīgā īpašība, t.i. tā nevar izbraukt, ja netiek pārvietotas divas citas mašīnas. (skat. https://en.wikipedia.org/wiki/Brute-force_search)
2) Autonomie (automātiskie) parkošanās algoritmi, kas mūsdienās aizvien vairāk tiek izmantoti automašīnās (skat. https://en.wikipedia.org/wiki/Automatic_parking)

Ļoti liels pēniecības darbu apjoms notiek sabiedriskā transporta, kas veidots no autonomajiem transportlīdzekļiem, sistēmu izstrādē. Viena no šo sistēmu priekšrocībām ir tā, ka parkošanos autonomie transportlīdzekļi var veikt ļoti efektīvi.

Šķīvji

Krievija



Kārtīgais bebrs Alise savus šķīvjus plauktā vienmēr sakārto pēc izmēriem tādā secībā, kā redzams attēlā augstāk: no kreisās puses, lielie šķīvji vispirms, tad vidējie un tad mazie. Alise nupat atklāja, ka ir aizmirsusi plauktā ielikt vienu lielo šķīvi.

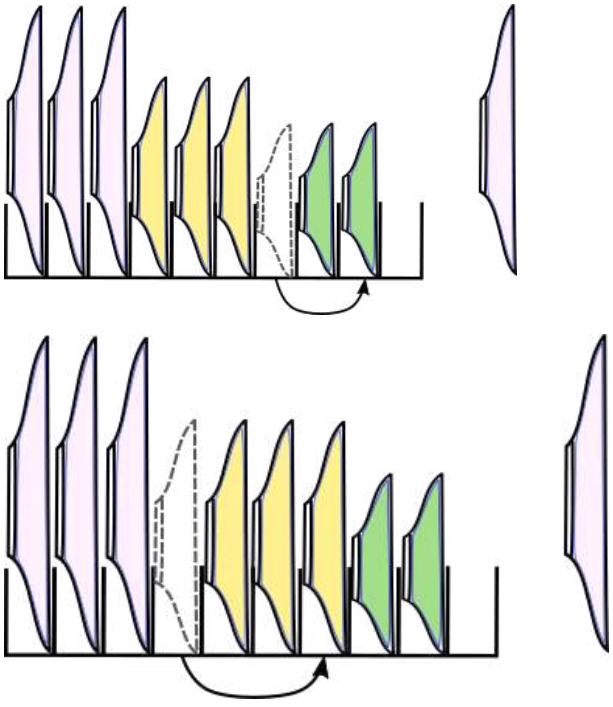
Uzdevums:

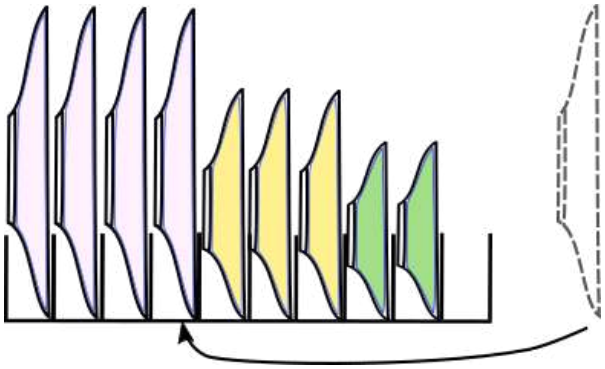
Kāds ir mazākais šķīvju skaits (ieskaitot jauno), kas jāpārvieto, lai šķīvjus sakārtotu ierastajā kārtībā?

Atbilžu varianti:

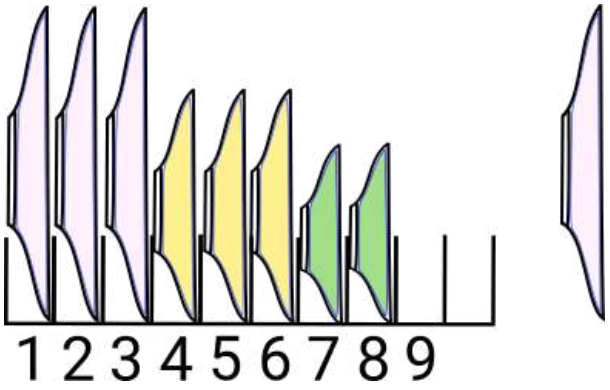
A. 3
B. 4
C. 5
D. 6

Pareizā atbilde: A





Uzdevumu var izpildīt, pārvietojot trīs šķīvjus. Iespējamais veids, kā to izdarīt, ir parādīts augstāk.



Lai nonāktu līdz šim risinājumam, mums vispirms jāsanumurē saliktie šķīvji augošā secībā no kreisās puses, un tad jāizvēlas labākā vieta papildu šķīvim. Lai visi lielie šķīvji būtu blakus, papildus šķīvim jāatrodas 4.vietā. Problēma ir tajā, ka mums 4.vietā jau atrodas vidējā izmēra šķīvis, tāpēc tam jāatrod jauna vieta. Jaunā vieta varētu būt 7.vietā, bet tajā jau atrodas mazais šķīvis, kas nozīmē, ka mazais šķīvis ir jāpārvieto no 7. uz 9.vietu.

Kā tas ir saistīts ar informātiku?

Acīmredzems šī uzdevuma atrisinājums ir pārvietot visus mazos un vidējos šķīvjus vienu vietu pa labi. Rezultātā tiks iegūta brīva vieta tieši tur, kur nepieciešams. Šādi rīkojoties, nāktos pārvietot sešus šķīvjus: vispirms šķīvi no 8. vietas uz 9., tad no 7. uz 8., utt. Šķīvju pārcelšana jāturpina līdz nākamais šķivis nav mazāks (ir tikpat liels vai lielāks) par to, kuru nepieciešams novietot plauktā. Tikai tad šķīvi varēs novietot vajadzīgajā vietā plauktā.

Aprakstīto algoritmu var lietot arī datoros, ar to atšķirību, ka parasti elementu skaits ir būtiski lielāks - pat vairāki miljardi. Līdz ar to viena elementa ievietošanai sakārtotā masīvā var nākties pārvietot miljardus elementu, kas, visticamāk, būs nepieņemami lēni. Lai paātrinātu šo procesu, datorzinātniekiem nākas izdomāt efektīvākus algoritmus, no kuriem viens tiek izmantots šī uzdevuma atrisināšanai.

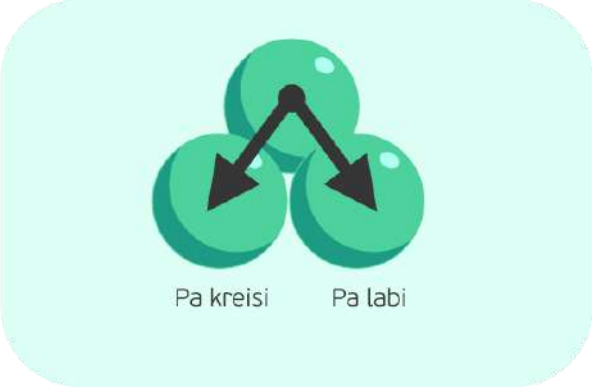
Ja elementu virknē ir vairāki vienādi elementi, un mēs zinām katra veida vienādo elementu skaitu, tad, lai ievietotu jaunu elementu, ir nepieciešams izdarīt tik gājienus, cik ir UNIKĀLU elementu, kas mazāki par ievietojamo + 1. Mūsu gadījumā ir divi šķīvju veidi (mazie un vidējie), kas mazāki par ievietojamo, tāpēc pareizās vietas atbrīvošanai nepieciešami divi gājieni. Papildus viens gājiens nepieciešams šķīvja novietošanai izbrīvētajā vietā.

Bumbu kaste

Beļģija

13 bumbas ir jāievieto trijstūra formas kastē, kā tas ir parādīts attēlā zemāk.

Ja mēs paceļam kastes augšējo stūri, brīvo vietu dēļ dažas bumbas ir briesmās - tām draud noripošana lejā.



Bumba ir briesmās, ja:

- Ir vismaz viena sprauga pa kreisi vai pa labi uzreiz zem tās;
- Vismaz viena bumba ir briesmās pa labi vai pa kreisi uzreiz zem tās.

Uzdevums:

Cik bumbas attēlotajā kastē NAV briesmās?

Pareizā atbilde: 8

Viens veids, kā tikt pie atbildes, ir sekojošs:

1. Saskaiti visas bumbiņas, kas IR briesmās;
2. No visu bumbiņu kopējā skaita atņem briesmās esošās bumbiņas, rezultātā iegūstot to bumbiņu skaitu, kas NAV briesmās.

Bet otrs risinājuma veids ir šāds:

1. Sāc ar apakšējo trijstūra rindu un atzīmē visas bumbas, kas tajā atrodas;
2. Turpini ar rindu augstāk;
3. Atzīmē visas bumbas tajā rindā, zem kurām atrodas jau iezīmētās bumbas;
4. Atkārto 2.un 3.soli, kamēr vairs nav palikusi neviena rinda;
5. Visas atzīmētās bumbas nav briesmās.

Kā tas ir saistīts ar informātiku?

Pēc uzdevumā dotā, bumba briesmās var būt divos gadījumos. Pirmo no tiem ir viegli pārbaudīt, bet otrajā nepieciešams noskaidrot, vai briesmās nav kāda bumba rindu zemāk. Lai gan izskatās pēc cirkulāras atsauces (“Bumba ir briesmās, ja bumba ir briesmās”), tik traki nav, jo bumbu kastes aplūkošanu varam sākt no apakšējās rindas. Tā kā zem tās citu rindu nav, tad varam apgalvot, ka šīs rindas bumbas nav briesmās, un varam aplūkot rindu augstāk un atzīmēt tās bumbas, kas ir briesmās. Šādi, virzoties uz augšu, varam atzīmēt visas briesmās esošās bumbas.

Bieži datoram doto instrukciju kopums ir līdzīgs tam, kādu dotu cilvēkam līdzīgu uzdevumu izpildei. Citkārt, instrukcijas var būt grūti izprotamas cilvēkam, bet rakstītas datoram piemērotā valodā. Sākumā minētā pieeja, kad definīcija atsaucas pati uz sevi, tiek saukta par rekursiju. Šajā gadījumā, definējot “bumba briesmās” kādai bumbai, tika izmantota “bumba briesmās” kādai citai bumbai, it kā šis stāvoklis mums jau būtu zināms.

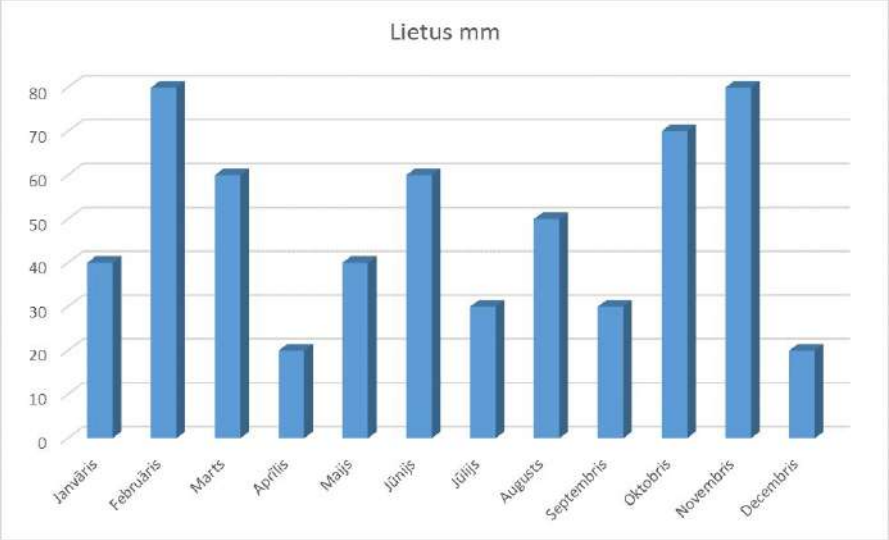
Cits piemērs ir definēt konkrēta cilvēka priekštečus. Skaidrs, ka tie ir vecāki, vecvecāki, vecvecvecāki, utt. Datorā ir grūti aprakstīt šādu bezgalīgu sakarību virkni, tāpēc datoram piemērotāka ir rekursīva definīcija: “Priekšteči ir vecāki un viņu priekšteči.” Lai rekursiju būtu iespējams izmantot, ir jābūt arī apstāšanās nosacījumam - piemēram, priekšteču gadījumā ir jāeksistē kādam “pēdējam priekštecim”, par kura vecākiem datu nav (lai gan realitātē šie vecāki bija!). Pretējā gadījumā priekšteču kopa būs bezgalīga, un tādu dators apstrādāt nevarēs.

Nokrišņu daudzums

Ungārija

Bebri plāno būvēt jaunu dambi. Balstoties uz ikgadējo nokrišņu daudzuma grafiku (attēlots zemāk), viņi cenšas atrast piemērotāko mēnesi tā uzbūvēšanai. Viņi ir vienojušies par sekojošiem kritērijiem:

- Viņi vēlas pārbaudīt dambja maksimālo ietilpību vislietainākajā gada mēnesī;
- Dambim jābūt uzbūvētam jau mēnesi vai divus pirms pārbaudīšanas, turklāt tam jānotiek iespējami sausākajā mēnesī.



Uzdevums:

Kurā mēnesī bebriem dambis jābūvē, lai ievērotu abus kritērijus?

Atbilžu varianti:

A) Janvārī
B) Aprīlī
C) Septembrī
D) Decembrī

Pareizā atbilde: D) Decembrī

Vispirms mums ir jāatrod lietainākie mēneši, kas ir februāris un novembris. Tā kā mēs zinām, ka dambis ir jāuzbūvē mēnesi vai divus pirms šiem mēnešiem, ir jāskatās uz decembri, janvāri, septembri vai oktobri. No šiem mēnešiem mums ir jāsameklē sausākais, kas šajā gadījumā ir decembris.

Kā tas ir saistīts ar informātiku?

Datorzinātnieku ikdienas darba sastāvdaļa ir lielākās, mazākās vai optimālās (kādā iepriekšzināmā nozīmē) vērtības meklēšana datos. Šī procesa laikā var būt jāņem vērā papildus nosacījumi, kas, turklāt, jāpielieto noteiktā secībā.

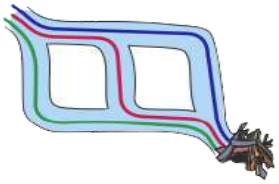
Diagrammas wiki: https://en.wikipedia.org/wiki/Diagram

Histogrammu wiki: https://en.wikipedia.org/wiki/Bar_chart

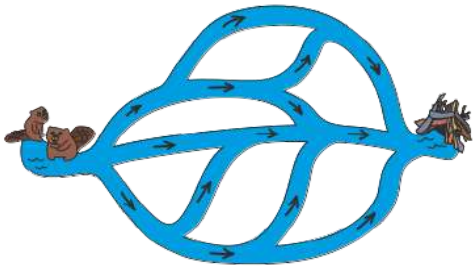
Bebru ceļi

Lietuva

Katram bebram vajag savu ceļu uz mājām, bet slinkuma dēļ pret straumi tie nepeldēs. Piemēram, attēlā redzamajā upes struktūrā var dzīvot 3 bebri, jo ir pieejami 3 ceļi no sākumpunkta līdz mājām.



Kāds bebru pāris dzīvo pie citas upes, kā parādīts šajā attēlā:

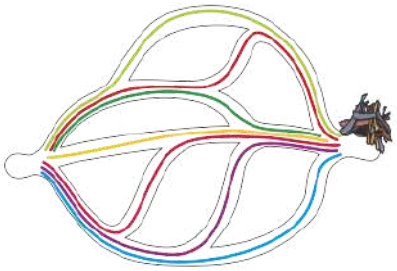


Uzdevums:

Bebru pāris vēlas rīkot viesības, uz kurām uzaicināt savus draugus. Kāds ir lielākais viesu skaits, kas var ierasties, lai katram viesim būtu savs ceļš (ņemot vērā, ka divi ceļi jau ir rezervēti bebru pārim)?

Pareizā atbilde: 5

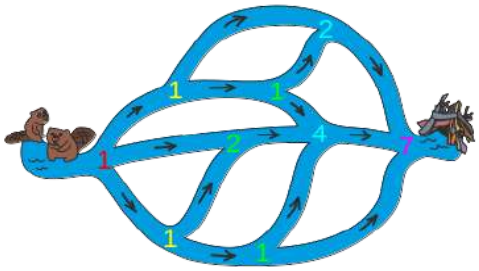
Ir 7 dažādi ceļi, kas ved uz bebru pāra mājām, 2 no tiem ir rezervēti jau viesību rīkotājiem. Tātad mēs varam skaitīt šos ceļus divos veidos:



Viens veids ir skatīties uz ceļiem attēlā un iezīmēt tos ar dažādām krāsām:

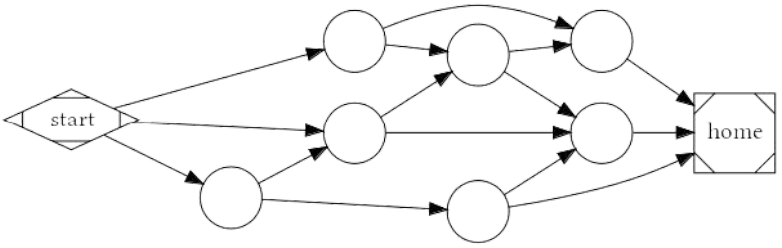
Otrs veids - izmantot dinamiskās programmēšanas metodi. Mēs skatāmies uz visiem attēla krustojumiem. Izskaitām, cik ceļi ved uz katru krustojumu un pierakstām atbilstošu ciparu tam. Lai to izdarītu, mēs ejam no kreisās uz labo pusi un saskaitām visus krustojumu numurus, kas ir tieši savienoti ar citu krustojumu ar bultām.

Šajā gadījumā mēs sākam ar 1 ielikšanu sākuma krustojumā (sarkanajā), jo ir tikai viens ceļš no sākotnējās bebru pozīcijas. Tas mēs izrēķinām ciparus dzeltenajiem krustojumiem - tie arī izveido skaitli 1, jo tie vienīgais iespējamais iepriekšējais krustojums ir sarkanais. Pēc tam mēs izrēķinām zaļos krustojumus: piemēram, lai iegūtu zaļo 2, mēs saskaitām sarkano 1 un dzelteno 1. Tālāk mēs izrēķinām zilos krustojumus, un tad pēdējo krustojumu: mēs iegūstam 7, saskaitot 2, 4 un 1 no iepriekšējiem krustojumiem.



Kā tas ir saistīts ar informātiku?

Dažādu uzdevumu atrisināšanu var veikt ērtāk un labāk, ja tiek izmantots noteikts datu attēlojuma veids. Viens no datu attēlošanas veidiem ir grafs. Uzdevumā doto ūdensceļu sistēmu var attēlot kā orientētu grafu, kur šķautnes atbilst upes posmiem, bet virsotnes - krustojumiem:



Orientēts grafs ir ļoti noderīgs tādos uzdevumos, kur nepieciešams noskaidrot, vai no vienas virsotnes ir iespējams nokļūt uz citu virsotni. Ja šķaunei piekārto skaitlisku vērtību (tas, piemēram, var atbilst attālumam starp objektiem dabā), tad iegūst svarotu grafu, kurā var meklēt ceļu starp divām virsotnēm ar īsāko svaru summu (atbilst īsākajam ceļam dabā).

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

Sistemātiska pieeja, kurā atrisinājums tiek būvēts izmantojot iepriekšajos soļos aprēķināto, tiek saukta par dinamisko programmēšanu. Dinamiskā programmēšana tiek plaši lietota dažādu uzdevumu risināšanā, bet to var izmantot tikai tad, ja katrs nākamais solis ir atkarīgs tikai no aktuālā stāvokļa, nevis no veida, kādā tas sasniegts.

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

Alerģijas

Slovēnija

Anna ballītei ir paredzējusi sagatavot ēdienus no sešām koku sugām, katra ēdiena pagatavošanai izmantojot vienas vai vairāku sugu kokus:

Osis Kļava Ozols

Bērzs Papele Vītols

Annai ir ballītes dalībnieku saraksts, kur katram bebram ir norādītas tās koku sugas, kuras šis bebrs var ēst (no pārējām ir alerģija).

Vārds	**Koks(i), kuru(s) var ēst**
Anna	Vītols, Ozols, Osis, Kļava
Toms	Vītols, Ozols, Papele
Sindija	Ozols
Dāvis	Osis, Bērzs
Emma	Vītols, Kļava, Bērzs
Fredis	Ozols, Osis
Jurģis	Papele, Kļava

Uzdevums:

Kāds ir mazākais ēdienu skaits, kas Anna jāsagatavo ballītei tā, lai katram ballītes dalībniekam būtu vismaz viens ēdiens, ko viņš var ēst?

Atbilžu varianti:

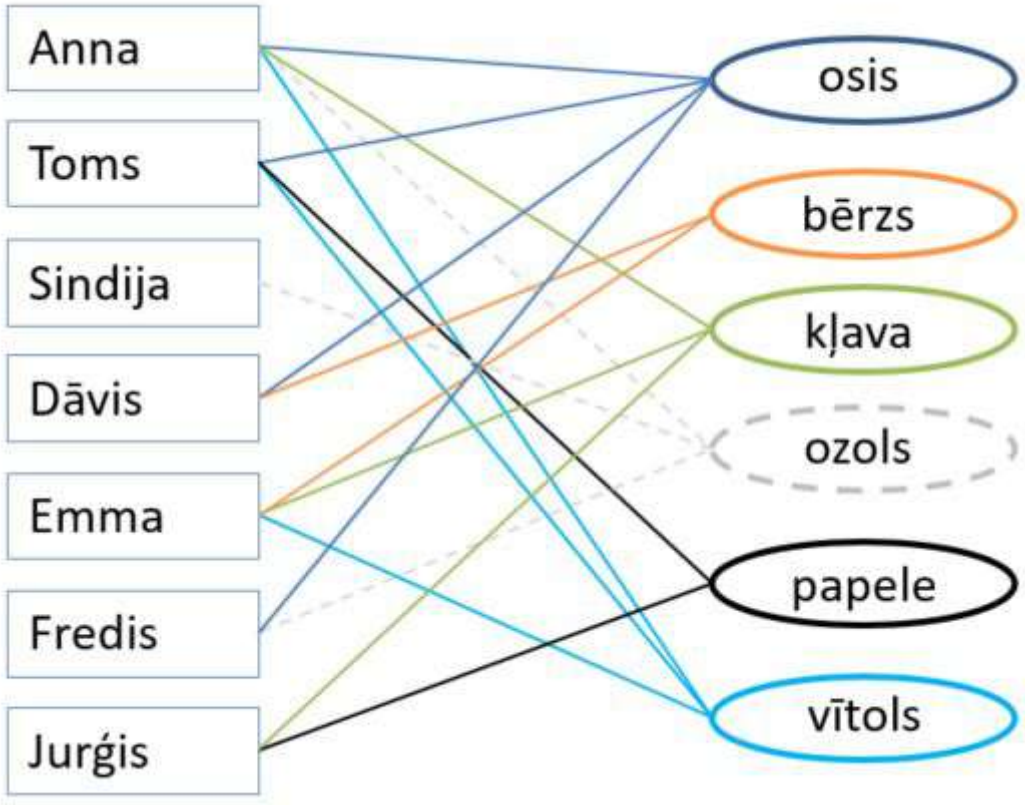
A) 1 B) 2 C) 3
D) 4 E) 5 F) 6

Pareizā atbilde: C

Annai noteikti ir jātaisa pusdienas no ozola Sindijai. Šīs pašas pusdienas derēs arī Annai, Tomam un Fredim. Nav nevienas kopīgas pusdienu iespējas atlikušajiem trim bebriem, tāpēc Annai būs jāgatavo vēl vismaz 2 ēdieni. Tas, par laimi, būs pietiekami: Anna var pagatavot osi un kļavu, papeli un bērzu vai kļavu un bērzu.

Kā tas ir saistīts ar informātiku?

Šis uzdevums ir klasisks divdaļīga grafa piemērs.
(https://en.wikipedia.org/wiki/Bipartite_graph)
Šajā uzdevumā vienu grafa daļu veido bebru kopa, bet otru - koku veidi.



Šis uzdevums algoritmu teorijā ir pazīstams kā dzelzceļa optimizācijas uzdevums, kurā dotiem vilcienu kustības grafikiem un pieturām nepieciešams atrast pēc iespējas mazāku staciju kopu, tā, lai katrs no vilcieniem pieturētu vismaz vienā no izvēlētajām stacijām (https://ipfs.io/ipfs/QmXoypizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Bipartite_graph.html).

Šo uzdevumu risina meklējot dominējošu kopu (https://ipfs.io/ipfs/QmXoypizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Dominating_set.html).

Iedomājieties sešas kopas, kas atbilst atšķirīgam ēdienam. Katrā kopā iekļausim tos bebrus, kam attiecīgais ēdiens garšo. Uzdevums ir izvēlēties tādu kopu (ēdienu) skaitu, ka šo kopu apvienojums (visi bebri, kam garšo vismaz viens no izvēlētajiem ēdieniem) satur visus elementus (bebrus). Citiem vārdiem, nepieciešams "pārklāt" visus elementus ar mazāko iespējamo kopu skaitu.

Līdzīgi kā tikko bebru un ēdienu uzdevums tika pārformulēts kopu valodā, daudzi citi, šķietami nesaistīti, uzdevumi var tikt "pārtulkoti" par to pašu (vai ļoti līdzīgu) abstraktu kopu uzdevumu.

Kopu pārklāšanas uzdevums ir pazīstams kā viens no grūtākajiem (NP-pilnajiem) datorzinātņu uzdevumiem. Šiem uzdevumiem nav zināmi efektīvi risināšanas algoritmi. Vispārīgs risināšanas algoritms paredz visu iespējamo variantu pilnu pārlasi, kas ir iespējama tikai ļoti maziem kopu izmēriem. Līdz ar kopas apjoma pieaugumu, iespējamo kombināciju skaits strauji aug un jau pie 240 elementiem dažādo variantu skaits ir salīdzināms ar atomu skaitu Visumā.

Kāpēc tad šo konkrēto uzdevumu tomēr izdevās atrisināt? Pirmkārt, kopu apjomi šeit bija ļoti mazi. Otrkārt, palīdzēja tas, ka Cecīlijai bija neliels izvēļu skaits. Visbeidzot, risināšanā tika izmantotas dažas vienkāršas loģiskas sakarības, kas noderēja šoreiz, bet varētu nebūt izmantojamas vispārīgā gadījumā.

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

Autobusu saraksts

Taizeme

Pilsētas autobusu satiksme ir organizēta divos maršrutos. Autobusu sarakstā redzams, kad katra maršruta autobuss pienāk katrā no pieturām.

Pietura	1. autobuss		
A	10:00	11:00	12:00
B	10:20	11:20	12:20
C	10:40	11:40	12:40
D	11:00	12:00	13:00
E	11:20	12:20	13:20

Pietura	2. autobuss	
A	10:10	11:10
F	10:20	11:20
C	10:30	11:30

Uzdevums

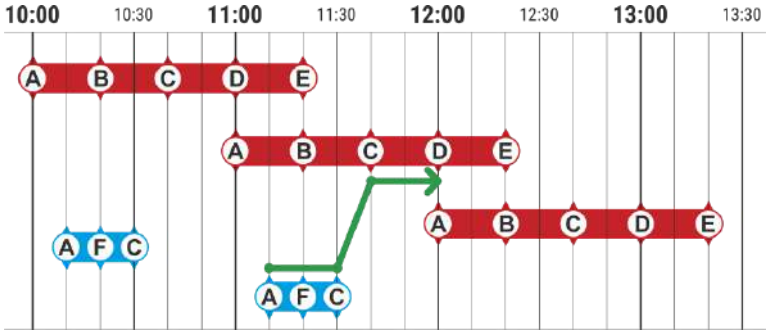
Ja bebrs Juris pieturā A ir plkst. 11:05, cikos viņš agrākais var sasniegt pieturu D?

Pareizā atbilde: 12:00

Bebrs Juris darīs šādi:

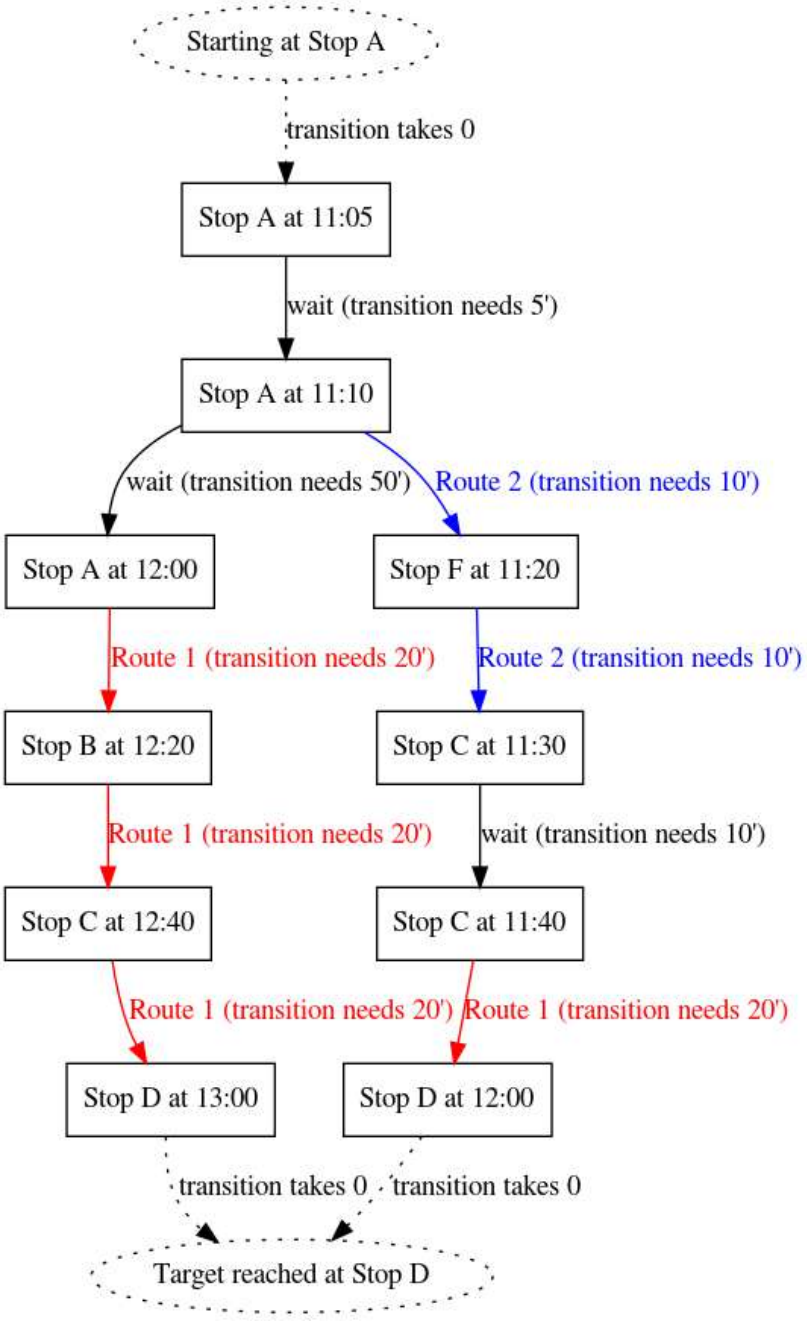
- Viņš dosies ar 2. maršruta autobusu no A pieturas 11:10 uz C pieturu 11:30
- Tad viņš izmantos 1. maršruta autobusu no pieturas C 11:40 uz pieturu D 12:00

Ja Juris dotos ar 1. maršruta autobusu no pieturas A 12.00, viņš pieturā D ierastos tikai 13:00



Kā tas ir saistīts ar informātiku?

Šī uzdevuma atrisināšanā izmantosim vienu no grafveida diagrammu paveidiem - stāvokļu diagrammu. Tās elementi būs stāvokļi - "kastītes", kurās ierakstīts diennakts laiks un pietura, kurā Juris atrodas, un pārejas no viena stāvokļa uz citu - "bultiņas", kas norāda šai pārejai nepieciešamo laiku. Šī uzdevuma diagramma (angļu valodā) izskatās šādi:



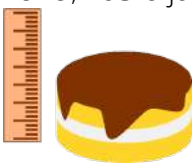
Tagad nepieciešams atrisināt īsākā ceļa atrašanas uzdevumu svarotā grafā - atrast ceļu (secīgu bultiņu virkni) no sākuma virsotnes (stāvoklis “Juris pieturā A”) līdz beigu virsotnei (stāvoklis “Juris pieturā D”) tā, lai šķautņu svaru (pārejas starp stāvokļiem laiku) summa būtu vismazākā.

Šoreiz no sākuma līdz beigu virsotnei ir divi ceļi - diagrammā pa kreisi ceļš ar summu 0+5+50+20+20+20+0=115, un pa labi - ar summu 0+5+10+10+10+20+0=55. Tādejādi, ceļš ar summu 55 ir ātrākais iespējamais. (atsauce: Shortest Path Problem)

Kūkas un kaimiņi

Beļģija

Piektdienas rītā trīs kaimiņienes - Anna, Betija un Klāra, sestdienas apkaimes ballītei katra pasūtīja pa kūkai no viena un tā paša konditora. Visas trīs sākotnēji pasūtīja viena tipa kūku, kas bija 3 cm augsta.



Tomēr katra pēcāk zvanīja konditoram atkārtoti, lai mainītu pasūtījumu. Katru reizi konditors pierakstīja jauno pasūtījumu, veco izmetot ārā. Kūkas būs gatavas agrā sestdienas rītā.

Anna zvana: Uztaisi manu kūku par 1cm augstāku, nekā es sākumā pasūtīju.

Vēlāk Anna zvana vēlreiz : Tagad uztaisi manu kūku tikpat augstu, kā Betijai.

Betija zvana: Uztaisi manu kūku par 2cm augstāku, nekā es sākumā pasūtīju.

Vēlāk Betija zvana vēlreiz: Uztaisi manu kūku par 1cm zemāku, nekā es pasūtīju.

Klāra zvana: Uztaisi manu kūku par 1cm augstāku, nekā Annas kūka.

Vēlāk Klāra zvana vēlreiz: Uztaisi manu kūku par vēl 1cm augstāku, nekā iepriekš pasūtīju.

PIEZĪME: Mēs nezinām, kādos laikos konditoram tika zvanīts. Zinām tikai to, ka otrais zvans no katras kaimiņienes tika veikts uzreiz pēc pirmā zvana.

Uzdevums:

Kurš no zemāk redzamajiem apgalvojumiem ir pareizs neatkarīgi no laikiem, kad zvani tika izdarīti?

Atbilžu varianti:

A) Annas un Betijas kūkām ir vienāds augstums
B) Betijas kūka ir vismaz 1cm zemāka par Klāras kūku
C) Klāras kūka ir tieši 2cm augstāka par Annas kūku
D) Visas 3 kūkas ir vismaz 4cm augstas

Pareizā atbilde: B

Atbilde A nav pareiza: ja Betija savus zvanus veica pēc Annas, Annas kūka beigās būs 3cm augsta, bet Betijas kūka - 4cm augsta. Tas uzreiz parāda, ka D nav pareizā atbilde.

Atbilde C nav pareiza: ja Klāra veiks savus zvanus pirmā, pēc tam Betija un tikai tad Anna, Klāras kūka būs 5cm augsta, Betijas kūka - 4cm, un Annas kūka arī 4cm.

Ja paskatāmies uz Betijas norādījumiem konditoram, var redzēt, ka kādā brīdī pēcpusdienā konditors būs pierakstījis augstumus 3cm, 5cm un 4cm Betijas kūkai. Tā rezultātā arī Annas kūkai cepējs varēja pierakstīt augstumus 3cm, 4cm un 5cm.

Tas nozīmē, ka sestdien Klāras kūka būs 5cm, 6cm vai 7cm augsta. Betijas kūka vienmēr būs 4cm augsta, tātad atbilde B ir pareizā.

Kā tas ir saistīts ar informātiku?

Mūsdienu datori var vienlaikus veikt vairākus darbus, tajā skaitā pat vienas lietojumprogrammas ietvaros. Piemēram, teksta apstrādes programmas var pārbaudīt pareizrakstību teksta ievades laikā.

Kā parādīts arī šajā uzdevumā, kad vairāki uzdevumi tiek izpildīti paralēli, ne vienmēr ir iespējams noteikt kāds būs iznākums. Tas tādēļ, ka bieži ir grūti paredzēt, kāda būs reālā darbu izpildes secība un kad kurš darbs reāli tiks izpildīts.

Programmēšanas veids, kurā vairākas programmas daļas var tikt izpildītas vienlaicīgi, sauc par laiksakritīgo programmēšanu. Rakstot šādas programmas, programmētājiem jāuzmanās no šādiem efektiem un par noteiktām daļām jābūt pārliecinātiem, ka tās vienmēr tiks izpildītas iepriekš paredzētajā secībā.

Piemēram, programmētājam, kurš izstrādā programmu tīkla printerim, jāpārliecinās, ka divi dokumenti, kas nosūtīti drukāšanai vienlaicīgi, tiks pilnībā drukāti viens aiz otra, nevis pamīšus, ņemot pa lapai (vai, vēl sliktāk, pa rindai!) no katra.

Ņemot vērā to, ka mūsdienu datori īpaši tiek veidoti vairāku vienlaicīgu uzdevumu izpildei, tādi šo iespēju korektas izmantošanas līdzekļi kā laiksakritīgā programmēšana iegūst aizvien lielāku nozīmi un tiek iekļauti datorzinātņu mācību programmās.

Lidojumu plānošana

Indija

Kad lidmašīna nolaižas lidostā, tai ir iedalīts atsevišķs skrejceļš no citām lidmašīnām, lai izvairītos no negadījumiem.

Bebru Salas Lidostā divas lidmašīnas nevar nolaisties uz viena un tā paša skrejceļa, ja to nolaišanās laiki atšķiras par 15 minūtēm vai mazāk.

Piemēram, 1.lidmašīna nolaižas plkst. 6.10, 2.lidmašīna plkst. 6.25, bet 3.lidmašīna plkst. 6.26. Šādā gadījumā 1.un 2.lidmašīna nevar izmantot vienu un to pašu skrejceļu, bet 3.lidmašīna var nolaisties uz tā paša skrejceļa, kuru izmantoja 1.lidmašīna.

Tu esi lidojumu kontrolieris, un tev jāpiešķir lidmašīnām skrejceļi pēc zemāk redzamā grafika:

Lidmašīna	**Laiks**
9W2400	7:00
9W1321	7:21
AI561	7:20
IP397	7:40
AI620	7:18
EC254	7:34
EK427	7:03
BQ439	7:36
SG147	7:12
VA035	7:28

Uzdevums:

Kāds ir mazākais skrejceļu skaits, kas nepieciešams, lai visas lidmašīnas nolaistos savos laikos atbilstoši noteikumiem?

Pareizā atbilde: 4

Vispirms lidmašīnas ar to nosēšanās laikiem jāsakārto augošā secībā:

1. 9W2400 7:00
2. EK427 7:03
3. SG147 7:12
4. AI620 7:18
5. AI561 7:20
6. 9W1321 7:21
7. VA035 7:28
8. EC254 7:34
9. BQ439 7:36

10. IP397 7:40

9W2400 nolaižas 7:00, tāpēc mēs tai dodam 1.skrejceļu. Nākamās EK427 un SG147 nolaižas ar nepilnu 15min atšķirību, tāpēc mēs katrai dodam savu skrejceļu. Tātad pašlaik mums ir izmantoti 3 dažādi skrejceļi.

Katrai nākamajai lidmašīnai mēs centīsimies piešķirt tos skrejceļus, kas jau ir izmantoti, ja tas ir iespējams. Turklāt vienmēr izvēlēsimies to skrejceļu, kas pirmais ir atbrīvojies - turot hronoloģiski vissenāk izmantotu skrejceļu "rezervē" un neļaujot uz tā nosēsties, kopējo nosēšanās grafiku var tikai pasliktināt. AI620 nolaižas 7.18, kas ir vairāk nekā 15min pēc pirmās lidmašīnas, tātad mēs to varam likt uz 1. skrejceļa. AI561 nevaram likt uz tā paša skrejceļa, jo nebūs pagājušas 15 minūtes no pirmā un ceturtā lidojuma, bet varam tai piešķirt 2. skrejceļu, jo tā ir par 15 minūtēm vēlāk nekā EK427.

9W1321 nolaižas 7:21, un tas ir mazāk par 15 min atšķirībā par SG147 (7:12), AI620 (7:18), un AI561 (7:20), kas jau ir sadalītas pa 3 skrejceļiem. Tātad mums šai lidmašīna ir jāizdala vēl cits skrejceļš. Un tā turpina ar visām lidmašīnām.

Tātad sanāk, ka lidmašīnas nolaidīsies uz šādiem skrejceļiem:

1.skrejceļš: 9W2400 (7:00), AI620 (7:18), EC254 (7:34)

2.skrejceļš: EK427 (7:03), AI561 (7:20), BQ439 (7:36)

3.skrejceļš: SG147 (7:12), VA035 (7:28)

4.skrejceļš: 9W1321 (7:21), IP397 (7:40)

Kā tas ir saistīts ar informātiku?

Šajā uzdevumā ir aprakstīti konflikti - divas lidmašīnas nedrīkst izmantot vienu skrejceļu, ja nolaišanās laiks atšķiras par ne vairāk kā 15 minūtēm) un resursi (piešķiramie skrejceļi), kas jāsadala tā, lai izvairītos no konfliktiem. Ir zināmi dažādi šī tipa uzdevumi. Piemēram, noskaidrot cik galdi būs nepieciešami saviesīgam pasākumam, lai pie viena galda nebūtu jāsēdina viesi, kas savā starpā nesatiek.

Pat ja aplūkojamo objektu (lidmašīnu, viesu, ...) skaits ir liels, jūs tik un tā vēlaties uzdevumu atrisināt ātri. Tādēļ informātikā šādu uzdevumu atrisināšanai ir izstrādātas īpašas metodes.

Viena no metodēm uzdevuma aprakstam izmanto grafu, kur virsotnes atbilst lidmašīnu reisiem, bet šķautnes - konfliktiem. Ja divu lidmašīnu reisu starpā ir konflikts (nolaišanās laiku mazās starpības dēļ tās nevar nolaisties uz viena skrejceļa), tad attiecīgās grafa virsotnes saista šķautne. Ja konflikta reisu starpā nav, tad arī grafā nav šķautnes starp attiecīgajām virsotnēm. Šādi sākotnējais uzdevums tiek reducēts uz grafa izkrāsošanas uzdevumu - ar kādu mazāko krāsu skaitu iespējams izkrāsot grafa virsotnes tā, lai neviena šķautne nesaistītu vienādi krāsotas virsotnes.

Viegli saprast, ka virsotnes krāsu var saistīt ar skrejceļa numuru sākotnējā uzdevumā.

Aprakstītajā uzdevumu klasē ietilpst dažādi resursu piešķiršanas uzdevumi. Piemēram, raidstaciju frekvenču piešķiršana, šablona atpazīšana, sporta spēļu plānošana, vietu plāna izveide, eksāmenu grafika sastādīšana.

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

Vienkāršākā šī uzdevuma risināšanas metode ir izmantojot t.s. “rijīgo” algoritmu - objektus (lidojumus) sakārtot augošā secībā pēc vērtības (to ielidošanas laika) un, sākot ar pirmo, pēc kārtas apstrādāt (piešķirt skrejceļus). Viegli pārliecināties, ka, lai optimāli izvietotu visus reisus, nepieciešams optimāli izvietot arī jebkuru mazāku hronoloģiski iepriekš notikušo reisu skaitu.

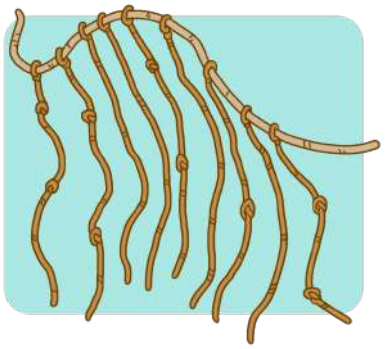
Lai rijīgais algoritms dotu optimālu risinājumu, nepieciešams pierādīt, ka katrā solī izvēloties variantu ar vislielāko vērtību un to pareizi apstrādājot, iegūtais risinājums būs labākais iespējams (vai viens no labākajiem).

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

Runājošie mezgli

Japāna

Lai paziņotu karaļvalsts jaunumus, karaliene izmanto “runājošos” mezglus, kas noteiktā veidā uzsieti uz virvēm. Piemēram, zemāk redzamie mezgli nozīmē: “svinam!”.



Nozīme ir tikai virvju secībai un katrā virvē iesieto meglu skaitam. Katrā virvē var būt iesieti 0, 1, 2 vai 3 mezgli. Ir tikai 50 dažādi iespējamie karalienes paziņojumi.

Uzdevums:

Kāds ir mazākais virvju skaits, kas karalienei ir nepieciešams jebkura paziņojuma attēlošanai?

Atbilžu varianti:

A) 2
B) 3
C) 4
D) 5

Pareizā atbilde: B

Ja būtu tikai viena virve, tad būtu iespējami tikai 4 dažādi paziņojumi, jo šai virvei būtu 0, 1, 2 vai 3 mezgli. Pievienojot 2. virvi, mums sanāku 4 iespējamie paziņojumi katrai, tātad 4x4=16 dažādi paziņojumi kopā. Šis aizvien nav pietiekami, tomēr, uzreiz kā 3.virve tiek pievienota, uzreiz jau ir 4x4x4=65 paziņojumi. Tā kā 65 ir vairāk par 50, šis ir pietiekami, lai parādītu visus iespējamos paziņojumus.

Kā tas ir saistīts ar informātiku?

Šajā uzdevumā jums nepieciešams domāt par skaitļu pozicionālo pierakstu. Runājošo mezglu gadījumā svarīgs ir virves novietojums un tajā iesieto mezglu skaits. Tā kā katrā virvē mazglu skaits var būt viena no četrām vērtībām, tad runājošo mezglu gadījumā mums ir darīšana ar četrinieku jeb kvarternāro skaitīšanas sistēmu.

Parasti, darbojoties ar naturāliem skaitļiem, tiek lietota decimālā skaitīšanas sistēma. Decimālie cipari (no 0 līdz 9) ir kā mezglu skaits šajā uzdevumā un katra cipara pozīcija (atrašanās vieta skaitļa pierakstā pēc kārtas) līdzinās virves novietojumam pēc kārtas un atbilst noteiktai desmitnieka pakāpei. Piemēram,

427=4x100+2x10+7 un ar trīs decimālajiem cipariem iespējams pierakstīt 10x10x10=1000 dažādus veselus nenegatīvus skaitļus (no 0 līdz 999).

Ļoti populāra ir arī divnieku jeb binārā skaitīšanas sistēma, kas tiek izmantota datoros datu un instrukciju glabāšanai. Šajā skaitīšanas sistēmā ir tikai divi dažādi cipari - 0 un 1, bet pozīcijas tiek sauktas par bitiem.

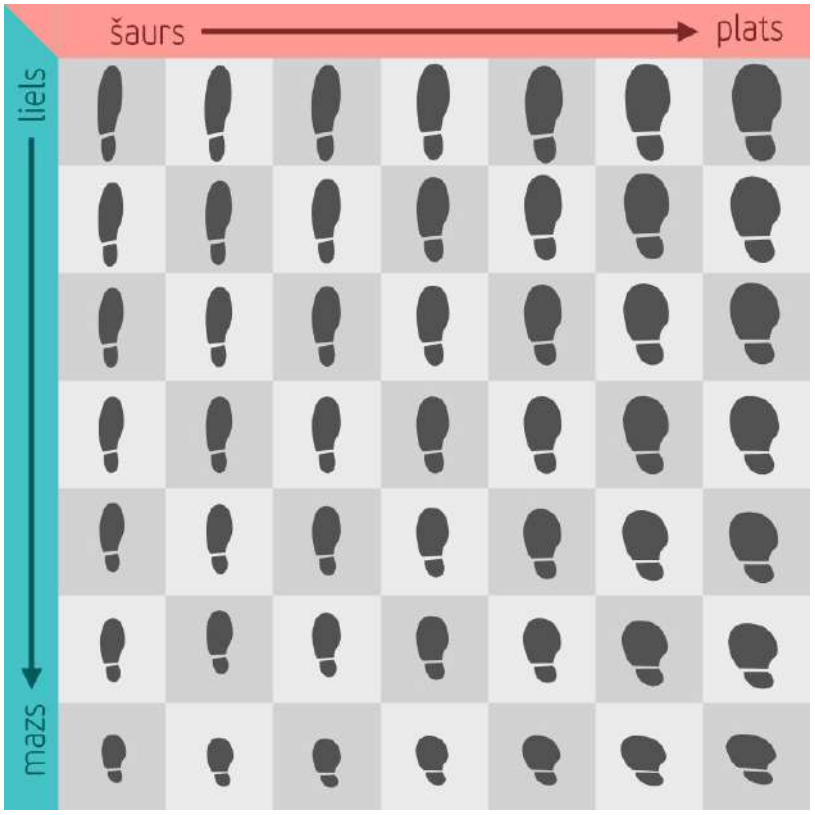
Īstais apavu izmērs

Dienvidkoreja

Bebrs devās uz apavu veikalu, lai nopirktu kurpju pāri. Viņš ieraudzīja vairākus apavus, kas bija sakārtoti tā, kā redzams attēlā zemāk.

Apavi bija sakārtoti augošā secībā pēc izmēriem un platumiem. Mazākā un šaurākā kurpe atradās apakšējā kreisajā stūrī, bet lielākā un platākā - augšējā labajā. Visi apavi bija atšķirīgos izmēros un platumos.

Būdams aizmāršīgs, bebrs ir aizmirsis savu apavu izmēru, tādēļ viņam būs jāmēra kurpes tik ilgi, kamēr atradīs tās, kas der. Pareizais pāris būs tas, kura izmērs un platums būs atbilstošs kājai. Pēc katras uzmērīšanas reizes bebrs saprot, vai uzmērītā kurpe pēc izmēra un/vai platuma der, un ja nē, vai nepieciešama mazāka vai lielāka un šaurāka vai platāka.



Bebrs izmanto metodi, kas garantē, ka ar n uzmērīšanas reizēm tiks atrasta īstā kurpe.

Uzdevums:

Kāds ir mazākais iespējamais n skaits?

Pareizā atbilde: 3

Bebram var paveikties atrast kurpi ar pirmo reizi, tomēr pareizo izmēru viņš var atrast divos piegājienos.

-Viņš var sākt ar pašu centrālo kurpi, kā parādīts attēlā zemāk:

**narrow: šaurs; wide: plats; big: liels; small: mazs; width: platums; size: izmērs*

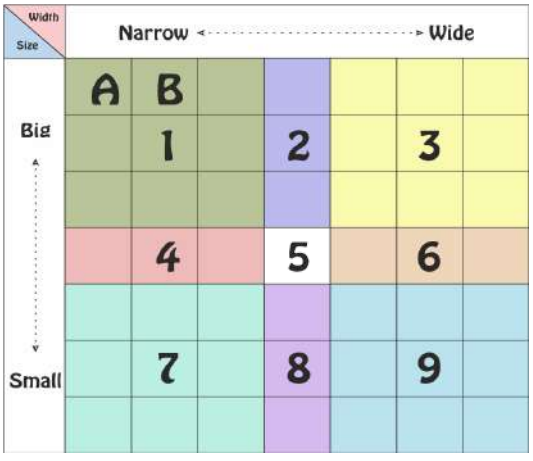
-Šī kurpe vai nu būs pareizajā izmērā, mazāka, lielāka, vai nu pareizajā platumā, šaurāka vai platāka. Balstoties uz piemērotību, bebrs sapratīs, vai kurpe viņam tieši derēs, vai būs kādā no zemāk esošajiem 9 krāsu lauciņiem;

-Ja kurpe viņam der, viņš atradis īsto pāri;

-Ja kurpe ir mazāka, bet platāka, viņam būtu jāmēra kurpes 1.zonā;

-Ja kurpe ir mazāka, bet īstajā platumā, viņam būtu jāmēra kurpes 2.zonā;

-Ja kurpe ir lielāka un šaurāka, tad jāmēra kurpes 9.zona un tā tālāk



Pieņemsim, ka bebra uzmērītās kurpes ir mazākas un platākas, nekā viņam der. Tādā gadījumā viņam ir jāmēra lielākas un šaurākas kurpes no 1.zonas.

Tagad viņam ir jāmēra centrālā kurpe zonā 1 (atzīmēta ar #1).

- Ja kurpe viņam der, viņš ir atradis īsto kurpju pāri
- Ja kurpe joprojām par mazu un platu, A kurpe būs īstā

- Ja kurpe būs par mazu, bet īstajā platumā, B kurpe būs īstā.

Kā redzam, bebram ir jāuzmēra kurpes vien 3 reizes, lai atradu sev nepieciešamo izmēru. Ja viņš mērīšanu sāks no jebkuras citas pozīcijas, jāmēra būs vairākkārt.

Kā tas ir saistīts ar informātiku?

Šajā uzdevumā kurpes veikalā ir sakārtotas augoši pēc izmēriem (vienā virzienā) un pēc platumiem (otrā virzienā). Meklēšanai sakārtotos datos efektīvi ir binārās meklēšanas algoritmi.

Šajos algoritmos ar katra vaicājuma palīdzību meklēšanas telpa tiek samazināta uz pusi, tādejādi ātri (logaritmiskā laikā) atrodot vajadzīgo.

Raksturīgs šādas meklēšanas piemērs ir pretinieka iedomāta skaitļa robežās no 1 līdz 100 atrašana, ja drīkst uzdot jautājumus formā “vai iedomātais skaitlis ir lielāks par N?” (kur N kāds skaitlis no 1 līdz 99)” un iespējamas tikai atbildes “jā” vai “nē”. Tad optimālā jautājumu uzdošanas stratēģija ir sākt ar N=50 un, atkarībā no saņemtās atbildes, katrā nākamajā gājienā izvēlēties atlikušā skaitļu segmenta viduspunktu.

Šī uzdevuma īpatnība ir tā, ka meklēšana notiek divdimensiju datos.

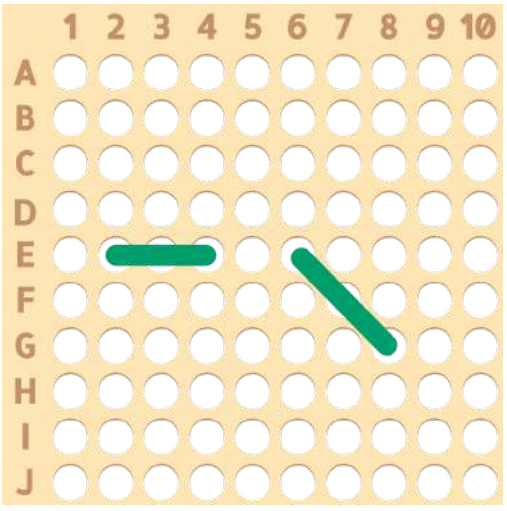
Izšuvums

Dienvidkoreja

Bebrs cenšas veidot izšuvumu, izmantojot speciālu programmu un mašīnu. Šī programma izmanto komandu ĀRĀ(cc)-IEKŠĀ(dd), kur cc un dd norāda uz adatas pozīciju režģī.

Piemēram, ĀRĀ(B2)-IEKŠĀ(A3) ir komanda, kas liek adatu novietot uz B2. Tad tā no aizmugures uz priekšu ir jāizvelk šajā lauciņā. Pēc tam adata ir jāpārvieto uz A4, kur tā ir jāiedur no priekšas uz aizmuguri.

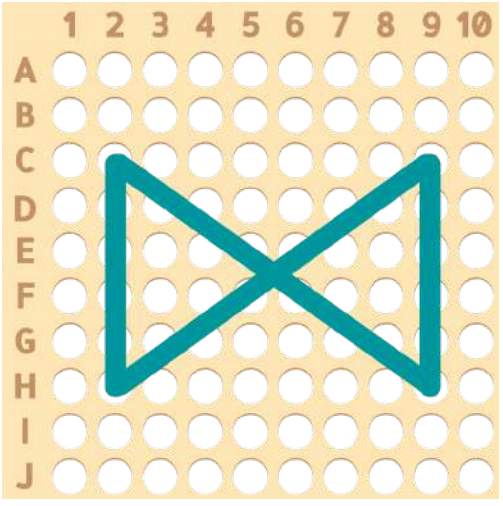
Zemāk redzamās komandas izveido šādu rakstu:



ĀRĀ(E6)-IEKŠĀ(G8);ĀRĀ(E2)-IEKŠĀ(E4)

Uzdevums:

Kādas komandas ir jāizmanto, lai izveidotu šādu rakstu?



A) ĀRĀ(H2)-IEKŠĀ(C2);ĀRĀ(H9)-IEKŠĀ(C9);ĀRĀ(C9)-IEKŠĀ(C2);ĀRĀ(H9)-IEKŠĀ(C2)

B) ĀRĀ(C2)-IEKŠĀ(H9);ĀRĀ(H2)-IEKŠĀ(C9);ĀRĀ(C2)-IEKŠĀ(H2);ĀRĀ(C9)-IEKŠĀ(H9)
C) ĀRĀ(H9)-IEKŠĀ(C9);ĀRĀ(H9)-IEKŠĀ(H2);ĀRĀ(C2)-IEKŠĀ(H2);ĀRĀ(C9)-IEKŠĀ(H2)
D) ĀRĀ(C2)-IEKŠĀ(C9);ĀRĀ(H2)-IEKŠĀ(H9);ĀRĀ(C2)-IEKŠĀ(H2);ĀRĀ(C9)-IEKŠĀ(H9)

Pareizā atbilde:

B)ĀRĀ(C2)-IEKŠĀ(H9);ĀRĀ(H2)-IEKŠĀ(C9);ĀRĀ(C2)-IEKŠĀ(H2);ĀRĀ(C9)-IEKŠĀ(H9)

Lai izveidotu attiecīgo rakstu, mums ir nepieciešamas 4 komandas - vienu katrai no četrām līnijām nenoteiktā secībā. Komandas ir

ĀRĀ(C2)-IEKŠĀ(H9) vai ĀRĀ(H9)-IEKŠĀ(C2)

ĀRĀ(H2)-IEKŠĀ(C9) vai ĀRĀ(C9)-IEKŠĀ(H2)

ĀRĀ(C2)-IEKŠĀ(H2) vai ĀRĀ(H2)-IEKŠĀ(C2)

ĀRĀ(C9)-IEKŠĀ(H9) vai ĀRĀ(H9)-IEKŠĀ(C9)

Izvēle, kurā bija visas četras komandas, bija atbildē B.

A nav pareizi, jo tajā ir komanda ĀRĀ(C9)-IEKŠĀ(C2), kas rada nevajadzīgu līniju. Tāpat arī komanda ĀRĀ(H2)-IEKŠĀ(C9) vai ĀRĀ(C9)-IEKŠĀ(H2) ir pazudusi.

B ir pareizā atbilde.

C nav pareizi, jo tajā ir komanda ĀRĀ(H9)-IEKŠĀ(H2), kas rada nevajadzīgu līniju. Tāpat arī komanda ĀRĀ(C9)-IEKŠĀ(H9) vai ĀRĀ(H9)-IEKŠĀ(C9) ir pazudusi.

D nav pareizi, jo tajā ir komanda ĀRĀ(C2)-IEKŠĀ(C9) un ĀRĀ(H2)-IEKŠĀ(H9), kas rada nevajadzīgas līnijas. Tāpat arī komanda ĀRĀ(C2)-IEKŠĀ(H9) vai ĀRĀ(H9)-IEKŠĀ(C2) un ĀRĀ(H2)-IEKŠĀ(C9) vai ĀRĀ(C9)-IEKŠĀ(H2) ir pazudusi.

Kā tas ir saistīts ar informātiku?

Algoritms apraksta uzdevuma paveikšanai izpildāmo soļu virkni. Lai gan jēdziens “algoritms” parasti saistās ar datorzinātni, tomēr tie ir atrodami arī ikdienas darbos. Īpaši, ja nepieciešams šos darbus veikt efektīvi. Šajā uzdevumā algoritma pielietošana ir demonstrēta izšūšanas mašīnai veidojot vēlamo rakstu.