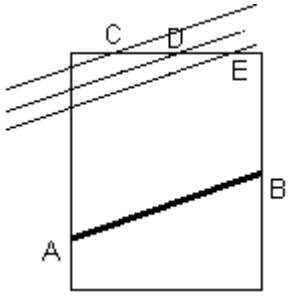
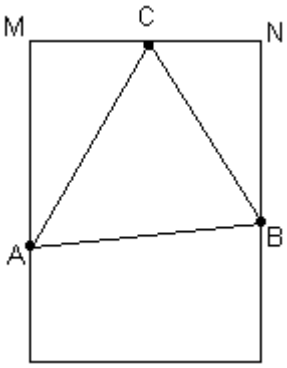
1. "LIELĀKAIS TRIJSTŪRIS"

Vienkārši ģeometriski apsvērumi ļauj izdarīt ļoti noderīgu secinājumu: ja uz kādas taisnstūra malas atrodas vairāk par diviem punktiem, tad visus punktus, izņemot divus malējos (kas atrodas vistuvāk šīs malas galiem), var izmest.



1. zīmējums



2. zīmējums

Vispirms aplūkosim gadījumu, kad visas lielākā laukuma trijstūra virsotnes atrodas uz dažādām malām. Paņemsim divus punktus A un B, kas atrodas uz pretējām taisnstūra malām. Aplūkosim visus iespējamos trijstūrus ar virsotnēm A un B, kuriem trešā virsotne atrodas uz trešās malas. 1. zīmējumā kā trešo virsotni var ņemt punktus C, D un E. Protams, vislielākais laukums trijstūrim ir tad, kad trešā virsotne atrodas iespējami tālāk no taisnes AB. Novilksim caur šiem punktiem taisnes, kas paralēlas taisnei AB, kā redzams zīmējumā. Taisne AB ir paliekta uz kreiso pusi, tāpēc vistālāk no tās atrodas punkts C, $S_{\Delta ABC} > S_{\Delta ABD} > S_{\Delta ABE}$. Ja taisne AB būtu paliekta uz labo pusi, tad vislielākais laukums būtu trijstūrim ABE. Ja taisne AB būtu novietota horizontāli, tad visiem trim trijstūriem būtu vienāds laukums, un kā vislielāko trijstūri mēs drīkstētu izvēlēties ΔABC. Jebkurā gadījumā punkts D ir lieks – punkta D izmešana nesamazina lielākā trijstūra laukumu.

Pierādīsim šo faktu stingrāk (skat. 2. zīmējumu).

Apzīmēsim nogriežņa MC garumu ar x. Tad

$$S_{\Delta ABC} = S_{AMNB} - S_{\Delta AMC} - S_{\Delta NBC} = S_{AMNB} - AM*x/2 - BN*(MN-x)/2 =$$
$$= (S_{AMNB} - BN*MN/2) + x*(BN-AM)/2.$$

Skaidrs, ka

- ja BN<AM, tad, palielinot x, iegūtā summa samazināsies, t.i., maksimālais laukums tiks sasniegts tad, kad uz malas MN tiek izvēlēts punkts, kas atrodas visvairāk pa kreisi;
- ja BN>AM, tad, palielinot x, iegūtā summa palielināsies, t.i., maksimālais laukums tiks sasniegts tad, kad uz malas MN tiek izvēlēts punkts, kas atrodas visvairāk pa labi;
- ja BN=AM, tad, mainot x, iegūtā summa nemainīsies, t.i., uz malas MN var izvēlēties jebkuru punktu, tai skaitā to, kas atrodas visvairāk pa kreisi.

Kad esam izvēlējušies punktu C, varam "pabīdīt" punktu A, nonākot pie secinājuma, ka arī virsotnei A ir jābūt uz attiecīgās taisnstūra malas malējai – pašai augšējai vai apakšējai. Analoģiski arī virsotnei B ir jābūt malējai.

Tagad aplūkosim gadījumu, kad lielākā laukuma trijstūra divas virsotnes, teiksim A un B, atrodas uz vienas malas. Trešo virsotni apzīmēsim ar C. $S_{\Delta ABC}$ vienāds ar AB un augstuma, kas vilkts no virsotnes C, reizinājuma pusi. Skaidrs, ka AB garums ir lielākais tad, ja A un B ir malējie punkti uz attiecīgās malas. Ja uz pretējās taisnstūra malas ir vismaz viens punkts, tad varam izvēlēties jebkuru no tiem – tas ar punktiem A un B veidos lielākā laukuma trijstūri. Protams, varam izvēlēties kādu no malējiem punktiem uz attiecīgās taisnstūra malas. Ja uz pretējās malas nav neviena punkta, tad kā trešo punktu jāizvēlas kāds no punktiem, kas atrodas uz kādas no blakusesošajām malām. Skaidrs, ka jāizvēlas punkts, kas atrodas iespējami tālāk no taisnes AB, t.i., malējo punktu uz vienas no blakusesošajām malām.

Tātad, pamatsecinājums ir – no visas doto punktu kopas uz katras taisnstūra malas jāatstāj ne vairāk kā divi punkti – katras malas malējie punkti.

Ja uz kādas no malām punktu nav, tad arī nav ko atstāt, bet, ja ir tikai viens punkts, tad to, protams, atstājam.

Rezultātā mums paliek ne vairāk kā 8 punkti. Atliek pārlasīt visus iespējamos trijstūrus ar virsotnēm šajos punktos un atrast lielāko no trijstūru laukumiem. Šī pārlase nav ilga – pavisam ir 56 veidi, kā izvēlēties trīs punktus no astoņiem.

Trijstūra laukuma aprēķināšanas paņēmieni, zinot tā virsotņu koordinātas, ir aprakstīti uzdevuma „Krāsainais trijstūris" atrisinājumā.

2. "AIZLIEGTĀS SUMMAS"

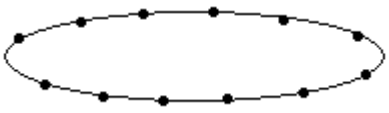
Padarīsim uzdevumu uzskatāmāku, izmantojot grafu. Dotie skaitļi būs grafa virsotnes. Savienosim divas virsotnes ar šķautni, ja virsotnēm atbilstošo skaitļu summa vienāda ar A vai B. Tagad uzdevums formulējams šādi: izvēlēties vislielāko iespējamo virsotņu skaitu tā, ka nekādas divas izvēlētās virsotnes nav savienotas ar šķautni.

Katra virsotne var būt savienota ar ne vairāk kā divām citām. Patiešām, visi dotie skaitļi ir atšķirīgi un, ja kādai virsotnei atbilst skaitlis x, tad šī virsotne var būt savienota tikai ar virsotnēm, kurām atbilst skaitļi (A-x) un (B-x). Viegli redzēt, ka izveidotajā grafā visas virsotnes sadalās saistītās grupās.

Iespējamas trīs veidu grupas:

1. izolēta virsotne – virsotne, kas nav savienota ar citām.

2. virsotņu ķēde:

3. vienkāršs cikls: 

Skaidrs, ka skaitļus, kas atbilst izolētām virsotnēm, Askoldam ir jāņem. Arī ar ķēdēm un cikliem viss ir skaidrs: ja ķēdē ir K virsotnes, tad Askoldam ir jāņem K/2 virsotnes, ja K ir pāra skaitlis, un (K+1)/2 virsotnes, ja K ir nepāra (citiem vārdiem, ((K+1) div 2) virsotnes jebkuram K). Ja ciklā ir K virsotnes, tad jāņem K/2 virsotnes,

ja K ir pāra, un (K-1)/2 virsotnes, ja K ir nepāra (citiem vārdiem, (K div 2) virsotnes jebkuram K).

Atlicis tikai sadalīt visas virsotnes saistītās grupās. Izvēlēsimies patvaļīgu virsotni, kas atbilst kādam skaitlim X (turpmāk teiksim – *virsotni X*).

Ja virsotne X nav savienota ne ar vienu citu virsotni, tad X ir izolēta virsotne – skaitlis X Askoldam ir jāņem, un virsotni X izmetam no virsotņu saraksta.

Ja virsotne X ir savienota tikai ar vienu virsotni Y, tad X ir ķēdes gala virsotne. Ejam pa šo ķēdi: atrodam Z – otru virsotni, ar kuru ir savienota virsotne Y, un tā tālāk, kamēr nonāksim līdz ķēdes otram galam – virsotnei, kas savienota tikai ar vienu virsotni. Visas K ķēdes virsotnes izmetam no virsotņu saraksta, bet Askolds ņem (K+1) div 2 virsotnes.

Ja virsotne X ir savienota ar divām virsotnēm, tad vai nu X ir iekšēja ķēdes virsotne, vai arī X ietilpst ciklā. Paņemsim vienu no divām virsotnēm, kas savienotas ar X, teiksim, virsotni Y, un turpināsim iet tajā pašā virzienā. Beigās būs viens no iznākumiem:

- Būsim atnākuši atpakaļ uz virsotni X. Tas nozīmēs, ka esam atraduši ciklu. Izmetam visas K atrastā cikla virsotnes no virsotņu saraksta, bet Askolds ņem K div 2 virsotnes.
- Būsim atnākuši uz virsotni, kas savienota tikai ar vienu virsotni. Tas nozīmēs, ka esam nonākuši ķēdes gala virsotnē, turklāt X ir šīs ķēdes iekšēja virsotne. Tad ejam uz citu virsotni, ar ko savienota virsotne X, un turpinām iet, kamēr atrodam virsotni, kas savienota tikai ar vienu virsotni, t.i., kamēr būsim nonākuši ķēdes otrajā galā. Izmetam visas K atrastās ķēdes virsotnes no virsotņu saraksta, bet Askolds ņem (K+1) div 2 virsotnes.

Atkal izvēlamies kādu no palikušajām virsotnēm un atkārtojam procesu, kamēr visas virsotnes būs izmestas.

Principā uzdevums ir atrisināts, bet atlicis to efektīvi realizēt. Būtībā jautājums ir – kā atrast visus savienotos virsotņu pārus. Ja vienkārši katrai virsotnei "skriesim cauri" visu virsotņu sarakstam, tad šis process prasīs $O(N^2)$ operāciju. Šāds risinājums ar dotiem uzdevuma ierobežojumiem ir pieļaujams, taču visas šķautnes var uzbūvēt arī ātrāk. Vispirms sakārtosim visus dotos skaitļus augošā secībā. Tagad, virzoties pretī no iegūtā saraksta abiem galiem, atradīsim visus skaitļu pārus, kuru summa vienāda ar A. Pseidokods izskatās šādi:

```
Kreisais:= 1; Labais:= N;
While Kreisais<Labais Do
  Case
  x[Kreisais]+x[Labais] < A: Kreisais:= Kreisais+1;
  x[Kreisais]+x[Labais] > A: Labais:= Labais-1;
  x[Kreisais]+x[Labais] = A: Fiksējam šķautni;
                             Kreisais:= Kreisais+1;
                             Labais:= Labais-1
  EndCase;
EndDo;
```

Skaitļu pāru, kuru summa vienāda ar B, meklēšana veicama analoģiski.

Katrs meklēšanas solis samazina starpību (Labais-Kreisais), tāpēc meklēšanas sarežģītība vienāda ar O(N), ja neskaita laiku, kas nepieciešams doto skaitļu sakārtošanai. Tā kā katra virsotne ir savienota ar ne vairāk kā divām citām, tad atrasto šķautņu glabāšanai varam izmantot divus masīvus: pieņemsim, ka x[i] ir i-tais skaitlis, bet connectA un connectB – palīgmasīvi; elementā connectA[i] ierakstīsim numuru skaitlim, kas summā ar x[i] dod A, bet elementā connectB[i] – numuru skaitlim, kas summā ar x[i] dod B. Ja kāds no šiem skaitļiem nav atrodams starp dotajām kartītēm, tad attiecīgajos elementos glabājam 0 (vai -1, vai kādu citu neiespējamu numuru).

Nav grūti redzēt, ka aprakstītās darbības, izņemot kārtošanu, prasa O(N) operāciju, tāpēc risinājuma sarežģītību nosaka kārtošanas sarežģītība. Būtu labi izmantot kādu efektīvu kārtošanas algoritmu, taču ar dotajiem ierobežojumiem (N<=1000) pietiekoši efektīva būs arī kārtošanas ar sarežģītību $O(N^2)$.

Rūpīgāka analīze ļauj nonākt pie secinājuma, ka uzdevumā virsotnes nevar veidot ciklu. Šo faktu nav grūti pierādīt, izmantojot doto faktu, ka uz kartītēm nav vienādu skaitļu, tāpēc tas paliek vingrinājums lasītājam. Tādējādi programma kļūst vienkāršāka – nav jāpārbauda gadījums, ka, sākot iet no virsotnes X, mēs atkal nonāksim tajā.

3. “MEGABŪDIŅAS”

Katrai no „stūra” figūrām ir viena raksturīga vieta – tās iekšējais stūris. Katrai no megabūdiņām ir savs raksturīgais 2x2 pikseļu fragments, kāds nav atrodams cita veida megabūdiņās:

Veids	ZA	ZR	DR	DA
Raksturīgais pikseļu izvietojums	# ␣ # #	␣ # # #	# # ␣ #	# # # ␣

Lai saskaitītu katra veida megabūdiņu skaitu, jāiziet cauri visam laukumam un jāpārbauda katrs kvadrāts ar 2x2 rūtiņām. Ja kvadrātā tieši trīs rūtiņās ir #, tas nozīmē, ka atrasts vienai megabūdiņai atbilstošs raksturīgais pikseļu fragments.

Rūtiņu laukumu ir vieglāk apstrādāt, ja laukumam apkārt izveido vienu pikseli platu tukšu (baltu) pikseļu rāmi.

4. “KRĀSAINAIS TRIJSTŪRIS”

1.posms. Vispirms atradīsim kādu trijstūri tā, lai katra tā virsotne būtu nokrāsota atšķirīgā krāsā. Katrai krāsai izvēlēsimies to punktu, kas failā dots kā pirmais. Pieņemsim, ka punkts A nokrāsots sarkanā, B – zilā, C – baltā krāsā. Ja šie punkti neatrodas uz vienas taisnes (kā to noteikt, aplūkosim vēlāk), tad vismaz vienu krāsainu trijstūri atrast ir izdevies un varam pāriet pie 2.posma. Ja punkti A, B un C atrodas uz vienas taisnes, tad pārlasām atlikušos punktus, līdz atradīsim kādu punktu K, kas neatrodas uz šīs taisnes. Tāds punkts noteikti atradīsies, jo pretējā gadījumā nebūtu spēkā uzdevuma nosacījumā teiktais: „Zināms, ka vismaz viens šāds (ar nenulles laukumu) trijstūris eksistē.” Nomainot to punktu no A, B un C, kurš ir tādā pat krāsā kā K, ar punktu K, būsim ieguvuši trīs punktus, kas katrs ir savā krāsā un veido korektu (kura laukums nav 0) trijstūri.

2.posms. Pēc pirmā posma beigām esam izvēlējušies trīs punktus A, B un C, kas veido korektu trijstūri. Ņemsim patvaļīgu dotu punktu D un pārbaudīsim, vai tas neatrodas ΔABC iekšpusē. Ja D atrodas uz ΔABC kontūra vai ārpus tā, pārejam pie nākamā punkta aplūkošanas.

Ja punkts D atrodas ΔABC iekšpusē, tad rīkosimies šādi: no punktiem A, B un C izvēlamies to punktu, kas ir tādā pat krāsā kā punkts D un nomainām šo punktu pret D. Ja pieņemam, ka D bija sarkanā krāsā – tad jānomaina punkts A un iegūsim trijstūri ΔBCD. Šim trijstūrim visas virsotnes ir atšķirīgās krāsās un tas pilnībā ietilpst ΔABC – tātad, ja kāds punkts X atrodas ārpus sākotnējā ΔABC, tad X atrodas arī ārpus Δ BCD. Bez tam punkts A atrodas ārpus ΔBCD, t.i., ΔBCD ir "labāks" par ΔABC – tā iekšpusē ir vismaz par vienu punktu mazāk.

Turpinām pārējo punktu aplūkošanas procesu, kā pamatu tagad jau izmantojot ΔBCD. Pie tam jau aplūkotie punkti otrreiz nav jāpārbauda – no iepriekšteiktā ir skaidrs, ka jau pārbaudītie punkti nevar atrasties ΔBCD iekšpusē.

Tātad pēc visu punktu secīgas pārbaudes un, ja nepieciešams, vienādas krāsas punktu nomaiņas, būsim ieguvuši meklēto trijstūri.

Atlicis atrisināt tehnisku apakšuzdevumu – noteikt, kur atrodas punkts D: ΔABC iekšpusē, uz tā kontūra vai ārpus tā. Parādīsim vairākus šī apakšuzdevuma atrisināšanas variantus.

1. Punkts D atrodas ārpus ΔABC tad un tikai tad, ja $S_{\Delta ABD} + S_{\Delta ACD} + S_{\Delta BCD} > S_{\Delta ABC}$. Šo apgalvojumu nepierādīsim. Trijstūra laukuma aprēķināšanai varam izmantot Hērona formulu:

$S = \sqrt{p \cdot (p-a) \cdot (p-b) \cdot (p-c)}$, kur a, b, c – trijstūra malu garumi, p – trijstūra pusperimetrs: p = (a+b+c)/2. Šādā veidā Hērona formula nav īpaši piemērota aprēķiniem, bet var atvērt iekavas un iegūt piemērotāku izteiksmi:

$$S = \frac{1}{2}\sqrt{2a^2b^2 + 2a^2c^2 + 2b^2c^2 - a^4 - b^4 - c^4}$$

Izteiksmes ērtums šeit izpaužas tādejādi, ka nav nepieciešams aprēķināt malu garumus, bet tikai to kvadrātus. Attāluma kvadrātu starp diviem punktiem $P(P_x, P_y)$ un $Q(Q_x, Q_y)$ viegli aprēķināt kā $d^2 = (P_x - Q_x)^2 + (P_y - Q_y)^2$.

Ja $S_{\Delta ABD} + S_{\Delta ACD} + S_{\Delta BCD} = S_{\Delta ABC}$, tad punkts D atrodas vai nu ΔABC iekšpusē, vai arī uz tā kontūra. Punkts D atrodas ΔABC iekšpusē tad un tikai tad, ja triju trijstūru laukumi $S_{\Delta ABD}$, $S_{\Delta ACD}$, $S_{\Delta BCD}$ ir lielāki par 0. Ja kaut viens no šiem laukumiem ir 0, tad punkts D atrodas uz ΔABC kontūra.

Trijstūra laukuma aprēķināšanas formulas var arī izmantot, lai noteiktu, vai trīs dotie punkti atrodas uz vienas taisnes. Trīs punkti A, B un C atrodas uz vienas taisnes tad un tikai tad, ja $S_{\Delta ABC} = 0$.

2. Trijstūra ABC laukuma aprēķināšanai var izmantot arī ievērojami vienkāršāku un praktiskāku formulu, kuru var izrēķināt ievērojami ātrāk un kuras aprēķināšanas laikā nerodas skaitliskas kļūdas, kas ir neizbēgamas, aprēķinot kvadrātsaknes:

$$S = \frac{1}{2}\left|(B_x - A_x) \cdot (C_y - A_y) - (C_x - A_x) \cdot (B_y - A_y)\right|,$$

kur (A_x, A_y) ir punkta A, (B_x, B_y) – punkta B, (C_x, C_y) – punkta C attiecīgi x un y koordinātas.

Var pat iztikt bez dalīšanas ar 2, bet visu laiku darboties ar divkāršotiem laukumiem – zinot, ka visu doto punktu koordinātas ir veseli skaitļi, šādi divkāršoti laukumi arī būs veseli skaitļi un visas darbības varēs veikt, izmantojot tikai vesela skaitļu tipa mainīgos.

Atgriežoties pie pārbaudes, vai trīs punkti A, B un C atrodas uz vienas taisnes, šajā gadījumā nosacījums $S_{\Delta ABC} = 0$ ir ekvivalents nosacījumam

$$(B_x - A_x) \cdot (C_y - A_y) = (C_x - A_x) \cdot (B_y - A_y)$$

3. Izteiksmes $\frac{1}{2}\left((B_x - A_x) \cdot (C_y - A_y) - (C_x - A_x) \cdot (B_y - A_y)\right)$ vērtība ir – orientēts trijstūra ABC laukums, ko apzīmēsim ar $L_{\Delta ABC}$. No iepriekšteiktā $|L_{\Delta ABC}| = S_{\Delta ABC}$, bet pati $L_{\Delta ABC}$ vērtība ir

- lielāka par 0, ja, apejot virsotnes pēc kārtas A→B→C→A, trijstūris ir apiets pretēji pulksteņrādītāja virzienam;
- mazāka par 0, ja, apejot virsotnes pēc kārtas A→B→C→A, trijstūris ir apiets pulksteņrādītāja virzienā;
- vienāda ar 0, ja punkti A, B un C atrodas uz vienas taisnes.

Punkts D atrodas trijstūra ABC iekšpusē tad un tikai tad, ja visi orientētie laukumi $L_{\Delta ABD}$, $L_{\Delta ACD}$ un $L_{\Delta BCD}$ vienlaicīgi visi ir vai nu pozitīvi, vai negatīvi.

Šis acīmredzot ir racionālākais aprēķināšanas veids.

4. Var meklēt taišņu AD un BC krustpunktu.

Ja šīs taisnes ir paralēlas, tad punkts D atrodas ārpus ΔABC.

Ja taisnes AD un BC krustojas punktā E, tad punkts D atrodas trijstūra ABC iekšpusē tad un tikai tad, ja vienlaikus ir spēkā šādi divi nosacījumi:

- punkts E atrodas starp punktiem B un C, un
- punkts D atrodas starp punktiem A un E.

Šo pierādījumu atstāsim lasītāju ziņā.

5. Caur punktu D novilksim horizontālu taisni un izanalizēsim, kā šī taisne krustos trijstūra ABC malas. Ja krustošanās vietu skaits ir 0 vai 1, tad punkts D atrodas trijstūra ABC ārpusē. Ja ir divi krustpunkti, tad punkts D atrodas trijstūra iekšpusē tad un tikai tad, ja tas atrodas tā nogriežņa iekšpusē, ko iegūst, šķeļot taisni un nogriezni.

Aplūkosim šo sakarību sīkāk.

Sakārtosim trijstūra ABC virsotnes pēc ordinātas: pieņemsim, ka $A_y <= B_y <= C_y$. Tad, ja $D_y >= C_y$ vai $D_y <= A_y$, tad punkts D atrodas vai nu ārpus trijstūra ABC, vai arī uz tā kontūra.

Ja $A_y < D_y < C_y$, tad horizontālā taisne, kas vilkta caur punktu D, šķērso trijstūra kontūru divos punktos. Atradīsim tos.

a. $D_y = B_y$. Šajā gadījumā viens krustpunkts ir punkts B, bet otrs – taisnes krustpunkts ar nogriezni AC.

b. $A_y<D_y<B_y$. Šajā gadījumā krustpunkti ir horizontāles krustpunkti ar nogriežņiem AB un AC.

c. $B_y<D_y<C_y$. Šajā gadījumā krustpunkti ir horizontāles krustpunkti ar nogriežņiem BC un AC.

Pēc tam, kad esam atraduši krustpunktus, atliek salīdzināt to abscisu vērtības ar D_x. Punkts D atrodas trijstūra iekšpusē tad un tikai tad, ja D_x ir lielāka par viena atrastā krustpunkta abscisas vērtību un mazāka par otru.

Algoritma sarežģītības novērtējums. Pirmajā posmā katra punkta piederība kādai taisnei tiek veikta ne vairāk kā vienreiz, bet pašai pārbaudei nepieciešams konstants laiks. Tādējādi, pirmajam posmam nepieciešamas O(N) operācijas, kur N – punktu kopskaits.

Otrajā posmā katram punktam (izņemot sākotnējā trijstūra virsotnes) vienu reizi tiek veikta šī punkta novietojuma pārbaude attiecībā pret kādu trijstūri. Katrai pārbaudei nepieciešams konstants operāciju skaits – t.i., arī otrajam posmam nepieciešamas O(N) operācijas.

Tādējādi, algoritma sarežģītība ir O(N) un atvēlētās atmiņas apjoms O(N), kas nepieciešams doto punktu glabāšanai.

5. "NEVIENNOZĪMĪGIE SKAITĻI"

Pats vienkāršākais risināšanas variants ir aplūkot visus naturālos skaitļus pēc kārtas un katram no tiem noskaidrot, vai tas ir neviennozīmīgs vai nē, un skaitīt, cik neviennozīmīgo skaitļu līdz šim atrasts. Skaidrs, ka, pārlasot pēc kārtas, kādreiz nonāksim arī līdz meklētajam skaitlim. Gandrīz tikpat viegli saprast, ka šāds risinājums lielām n vērtībām būs ārkārtīgi lēns ($O(a_n)$).

Mazliet labāku risinājumu var iegūt, ja ievēro, ka visi skaitļi, kas satur kādu no cipariem 3, 4 vai 7, **vienmēr ir viennozīmīgi**. Līdz ar to pietiek aplūkot nevis visus skaitļus pēc kārtas, bet tikai „apgriežamos" skaitļus, kas sastāv no cipariem 0, 1, 2, 5, 6, 8, 9 (un tāpēc arī apgrieztais skaitlis sastāvēs tikai no šiem cipariem). Nākamais apgriežamais skaitlis šajā gadījumā jāmeklē nevis vienkārši pieskaitot 1, bet katrā pozīcijā jāmeklē nākamais cipars, zinot, ka aiz 0 seko 1, aiz 1 – 2, aiz 2 – 5, aiz 5 – 6, aiz 6 – 8, aiz 8 – 9, aiz 9 – 0. Pēdējā gadījumā veidojas pārnese. Līdz ar to pārlasāmo skaitļu skaits kļūst mazāks, bet iepriekš aprakstītajā veidā veikta pārlase pēc kārtas katram skaitlim veicot „viennozīmīguma" pārbaudi, joprojām būs lēna ($O(a_n^{\lg 7})$).

Abi iepriekšējie risinājumi, meklējot n-to neviennozīmīgo skaitļu virknes locekli, pa ceļam atrod arī visus iepriekšējos n-1 virknes locekļus.

Efektīvu risinājumu var iegūt, ja nemēģina atrast **visus** virknes locekļus.

No iepriekšteiktā ir redzams, ka pavisam ir 7^n apgriežamas n-ciparu virknes, kas sastāv tikai no cipariem 0, 1, 2, 5, 6, 8 vai 9 un tāpēc, iespējams, ir neviennozīmīgs skaitlis. Protams, starp šīm virknēm ir arī tādas, kas nav neviennozīmīgs skaitlis – piemēram, virknes, kas sākas vai beidzas ar 0. Katram n apzīmēsim šādu n-ciparu apgriežamu virkņu skaitu ar V_n.

Lai atrastu neviennozīmīgo n-ciparu skaitļu skaitu P_n (n>2), rīkosimies šādi – aplūkosim divciparu skaitli, ko veido kāda apgriežama n-ciparu skaitļa pirmais un pēdējais cipars. Ir iespējams, ka šis skaitlis ir neviennozīmīgs. Tad atlikušajiem n-2 cipariem nav īpašu prasību un der visas apgriežamās virknes garumā n-2 (tādu ir V_{n-2}). Savukārt, ja divciparu skaitlis ir viennozīmīgs, tad atlikušajiem n-2 cipariem jāveido neviennozīmīga apgriežama virkne (atšķirībā no neviennozīmīgiem skaitļiem, šāda neviennozīmīga virkne var saturēt 0 sākumā un/vai beigās).

$$P_n = P_2V_{n-2} + Q_2P'_{n-2}$$

Ar Q_n tiek apzīmēts n-ciparu viennozīmīgo skaitļu skaits.

Ar P'_n un Q'_n tiek apzīmēts attiecīgi neviennozīmīgo un viennozīmīgo apgriežamo virkņu skaits garumā n (atšķirībā no skaitļiem, nav ierobežojuma uz 0 lietošanu).

Katram n ir spēkā $P'_n + Q'_n = V_n$.

Varam atrast formulas arī Q_n, P'_n un Q'_n aprēķināšanai:

$$Q_n = Q_2Q'_{n-2}$$

$$P'_n = P'_2V_{n-2} + Q'_2P'_{n-2}$$

$$Q'_n = Q'_2Q'_{n-2}$$

Lai varētu veikt šos aprēķinus katrai n vērtībai, vēl ir nepieciešamas šādas vērtības:

$P_1 = 2$ (skaitļi 6 un 9)

$Q_1 = 4$ (skaitļi 1, 2, 5, 8)

$P'_1 = 2$ (skaitļi 6 un 9)

$Q'_1 = 5$ (skaitļi 0, 1, 2, 5, 8)

$Q_2 = 6$ (skaitļi 11, 25, 52, 69, 88, 96)

$P_2 = 30$ (visi apgriežamie divciparu skaitļi, kas nesatur 0, izņemot tos, kas pieder Q_2)

$Q'_2 = 7$ (virknes 00, 11, 25, 52, 69, 88, 96)

$P'_2 = 42$ (visas apgriežamās virknes garumā 2, izņemot tās, kas pieder Q'_2)

Tālākajos aprēķinos būs nepieciešamas arī sakarības $V_0=Q'_0=1$, $P_0=Q_0=P'_0=0$

Zinot P_n vērtības, iespējams noteikt, cik ciparu skaitlis ir a_n. Ja $n \le P_1$, tad a_n ir viencipara, pretējā gadījumā, a_n ir vairāk par vienu ciparu.

Ja $n-P_1 \le P_2$, tad a_n ir divciparu, pretējā gadījumā a_n ir vairāk par diviem cipariem, utt.

Lai noskaidrotu a_n ciparu skaitu c, no sākotnējā n pēc kārtas atņemsim P_1, P_2, ..., P_{c-1} tik ilgi, kamēr atņemšanas rezultāts ir pozitīvs skaitlis. Atņemšanas procesā atlikušais skaitlis norāda, kurš skaitlis starp visiem c-ciparu skaitļiem pēc kārtas ir a_n (apzīmēsim to ar d: $1 \le d \le P_c$).

Kad noskaidrotas c un d vērtības, atliek atrast pašu neviennozīmīgo skaitli.

Tas tiks veikts, noskaidrojot pēc kārtas a_n ciparu vērtības. Pats mazākais c-ciparu skaitlis ir 10...0 (c-1 nulle). Mazākais iespējamais skaitļa pirmais cipars ir 1,

tāpēc noskaidrosim, cik ir tādu dažādu neviennozīmīgu skaitļu garumā c, kas sākas ar 1. Ja skaitļa pēdējais cipars arī ir 1, tad „viennozīmīgumu" noteiks skaitļa vidus fragments (bez pirmā un pēdējā cipara) garumā c-2. Pēc iepriekš aprēķinātajām sakarībām šādu skaitļu skaits ir P'_{c-2}.

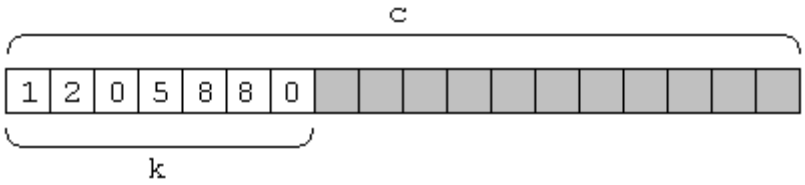
Ja pēdējais cipars ir atšķirīgs no 1, tad skaitlis, neatkarīgi no vidus fragmenta satura (tam, protams, ir jābūt apgriežamam skaitlim, tāpēc pēdējais cipars nevar būt 0) būs neviennozīmīgs. Šādu skaitļu kopskaits ir V_{c-2}. Tātad c-ciparu neviennozīmīgo skaitļu, kas sākas ar 1, skaits ir $P'_{c-2}+5V_{c-2}$.

Ja d nepārsniedz šo skaitli, tad a_n pirmais cipars tiešām ir 1 un atliek atrast nākamos ciparus. Pretējā gadījumā meklētā skaitļa pirmais cipars ir lielāks par 1. No d ir jāatņem c-ciparu neviennozīmīgo skaitļu, kas sākas ar 1, skaits un kā nākamais kandidāts jāapskata nākamais apgriežamais cipars. Apskatot 2, un, ja nepieciešams, arī 5, 6, 8 un 9 pēc kārtas kā pirmā cipara kandidātus, atradīsim īsto sākuma ciparu.

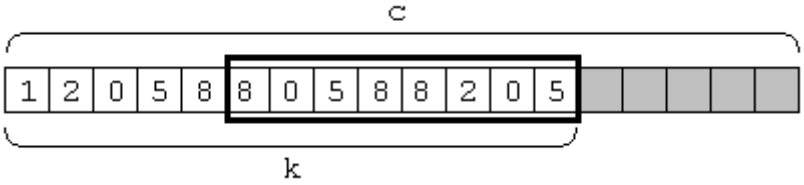
Kad noteikts pirmais cipars, varam meklēt otro un nākamos ciparus, izmantojot līdzīgu pieeju kā pirmā cipara noteikšanas laikā.

Pieņemsim, ka līdz šim izdevies noteikt garāku skaitļa sākuma fragmentu, kas sastāv no vairākiem (k, k>1) cipariem. Šo sākuma fragmentu nosauksim par **prefiksu** un aprēķināsim, cik ir tādu neviennozīmīgo c-ciparu skaitļu, kas sākas ar šo prefiksu.

Ja k≤c/2, tad atkal šķirosim divus gadījumus – ja beigu fragments garumā k ir apgriezts prefikss, tad šādu neviennozīmīgo skaitļu skaits ir P'_{c-2k}. Ja beigu fragments garumā k nav apgriezts prefikss, tad vidus fragments var būt jebkāda apgriežama virkne. Pavisam kopā šādu virkņu ir $(6*V_{k-1}-1)*V_{c-2k}$.



Ja k>c/2, tad jānoskaidro, vai skaitļa vidus fragments garumā 2k-c ir neviennozīmīgs vai nē. Ja šis vidus fragments ir neviennozīmīgs, tad beigu fragments garumā c-k var būt jebkura apgriežama virkne, kas nebeidzas ar 0. Šādu skaitļu pavisam ir $6V_{c-k-1}$. Ja vidus fragments garumā 2k-c nav neviennozīmīgs, tad nepieciešams, lai šo „neviennozīmīgumu" nodrošinātu beigu fragments garumā c-k, kas nedrīkst sakrist ar tāda paša garuma sākuma fragmenta apgriezto. Šādu skaitļu skaits ir $6V_{c-k-1}-1$ (visi fragmenti garumā c-k, kas nebeidzas ar 0, izņemot vienu, kas neder).



Īpašs gadījums ir k=c-1, kas nozīmē, ka atlicis noskaidrot pēdējo ciparu. Šajā gadījumā pietiek vienkārši virzīties uz priekšu, ņemot nākamo apgriežamo skaitli un to pārbaudot (kā aprakstīts neefektīvajā risinājumā).

Šāda atrisinājuma sarežģītība ir $O(\log(a_n))$.

6. “LEJASBEBRU CIEMA KARTE”

Izmantosim to pašu ideju kā uzdevumā „Megabūdiņas”.

Par *iekšēju stūri* kartē sauksim jebkuru 2x2 pikseļus lielu kvadrātu, kurā trīs pikseļi ir melni un viens balts. Pavisam ir četra veida iekšēji stūri:

Veids	1	2	3	4
Pikseļu izvietojums				

Katrai "L" formas mājai ir viens "1" veida stūris,

katrai "T" formas mājai ir viens "3" un viens "4" veida stūris,

katrai "C" formas mājai ir viens "1" un viens "4" veida stūris,

katrai "M" formas mājai ir divi "3" un divi "4" veida stūri,

katrai "O" formas mājai ir pa vienam katra veida stūrim.

Ar l, t, c, m un o apzīmēsim attiecīgi "L","T","C","M" un "O" māju skaitu, bet ar a1, a2, a3, a4 — attiecīgā veida iekšējo stūru skaitu kartē. Saskaitīt iekšējo stūru skaitu nav sarežģīti – jāiziet cauri visam laukumam un jāpārbauda katrs kvadrāts ar 2x2 rūtiņām. Ja kvadrātā tieši trīs rūtiņās ir #, tas nozīmē, ka atrasts iekšējs stūris.

Zinot katras formas stūru skaitu katras formas mājā, izveidosim vienādojumu sistēmu:

$$l+c+o = a_1$$

$$o = a_2$$

$$t+2m+o = a_3$$

$$t+c+2m+o = a_4$$

Uzreiz iespējams atrast $o = a_2$. Atņemot no ceturtā vienādojuma trešo, iegūsim $c = a_4-a_3$. No pirmā vienādojuma: $l = a_1-c-o = a_1-a_4+a_3-a_2$, bet no trešā iegūsim $t+2m = a_3-a_2$. Ar iepriekšminētajām sakarībām nepietiek, lai aprēķinātu t un m vērtības.

Lai tās aprēķinātu, nepieciešams saskaitīt viena noteikta veida ārējus stūrus (2x2 pikseļus lielus kvadrātus ar trim baltiem un vienu melnu pikseli):

"T" formas mājām ir divi šādi stūri, "M" formas mājām – trīs, "L", "C" un "O" formas mājām – pa vienam. Apzīmējot šādu ārēju stūru skaitu ar a_5, iegūsim vēl vienu vienādojumu:

$$2t+3m+l+c+o = a_5$$

No šejienes: $2t+3m = a_5-l-c-o = a_5-a_1$.

Ņemot vērā iepriekš atrasto sakarību $t+2m = a_3-a_2$, iegūsim, ka $m = a_1-2a_2+2a_3-a_5$, $t = -2a_1+3a_2-3a_3+2a_5$.