Apakštaisnstūri

Šim uzdevumam eksistē acīmredzams un neefektīvs risinājums - ciklā izvēlamies taisnstūrveida apgabala kreiso augšējo un labo apakšējo rūtiņu, aprēķinām skaitļu summu šajā apgabalā un, ja tā atrodas vajadzīgajās robežās, tad pieskaitām derīgo taisnstūru skaitam 1.

Efektīvam risinājumam pārlase jāorganizē gudrāk.

Vispirms, jau ielasot, glabāsim nevis pašas tabulas vērtības, bet skaitļu summu sākot no rindas sākuma. Uzdevumā dotajam piemēram ielasītā tabula būtu šāda:

1	1	3	6
6	13	13	16
3	3	7	9

Pēc tam saturīgi notiek tas, kas tika izmantots neefektīvajā risinājumā - izvēlamies tabulas sākuma rindu (cikls no 1 līdz N) un aplūkojam visas iespējamās tabulas beigu rindas (cikls no sākuma rindas līdz N).

Šī pārlase tiek organizēta izmantojot viendimensiju masīvu v, kurā sākumā ielasa un sākuma rindas vērtības (kolonnu numerācija no 1, bet $v_0 = 0$), bet pēc tam šim masīvam pakāpeniski pieskaita nākamo rindu saturu.

Ciklā tiek aplūkota sākuma kolonna k un tiek noskaidrots, cik elementiem šajā viendimensiju masīvā summas vērtība ir derīga.

Katrā brīdī tiek izmantoti divi rādītāji: r_A - pirmās kolonnas indekss, kur elementu summa ir vismaz $A + v_{k-1}$, un r_B - pirmās kolonnas indekss, kur elementu summa ir lielāka par $B + v_{k-1}$. Tad derīgo apgabalu skaits sākuma kolonnai k ir vienāds ar $r_B - r_A$. Svarīgi ir pamanīt, ka palielinot sākuma kolonnas k vērtību r_A un r_B vērtības nevar samazināties. Tas nozīmē, ka masīva v elementi (kopskaitā M) ar rādītāju palīdzību tiek caurskatīti vienreiz un šajā mehānismā nav ieslēpti papildus cikli.

Aplūkosim, kā šis algoritms darbojas uzdevuma tekstā dotajā piemērā (N=3, M=4):

1) `sākuma_rinda=1, beigu_rinda=1`

v[0]	v[1]	v[2]	v[3]	v[4]
0	1	1	3	6

$k = 1 \Rightarrow r_A = 5, r_B = 5$ Tā kā $r_A > M$, tad šai beigu rindai derīgu taisnstūru nebūs - ejam ārā no kolonnu cikla.

2) `sākuma_rinda=1, beigu_rinda=2`

v[0]	v[1]	v[2]	v[3]	v[4]
0	7	14	16	22

$k = 1 \Rightarrow r_A = 1, r_B = 2$ Derīgo variantu skaits 2-1=1.
$k = 2 \Rightarrow r_A = 2, r_B = 4$ Derīgo variantu skaits 4-2=2.
$k = 3 \Rightarrow r_A = 4, r_B = 5$ Derīgo variantu skaits 5-4=1.
$k = 4 \Rightarrow r_A = 5, r_B = 5$ Tā kā $r_A > M$, izeja no kolonnu cikla.

3) `sākuma_rinda=1, beigu_rinda=3`

v[0]	v[1]	v[2]	v[3]	v[4]
0	10	17	23	31

$k = 1 \Rightarrow r_A = 1, r_B = 1$ Derīgo variantu skaits 1-1=0.
$k = 2 \Rightarrow r_A = 2, r_B = 3$ Derīgo variantu skaits 3-2=1.

$k = 3 \Rightarrow r_A = 4, r_B = 4$ Derīgo variantu skaits 4-4=0.
$k = 4 \Rightarrow r_A = 4, r_B = 5$ Derīgo variantu skaits 5-4=1.

4) `sākuma_rinda=2, beigu_rinda=2`

v[0]	v[1]	v[2]	v[3]	v[4]
0	6	13	13	16

$k = 1 \Rightarrow r_A = 2, r_B = 2$ Derīgo variantu skaits 2-2=0.
$k = 2 \Rightarrow r_A = 2, r_B = 4$ Derīgo variantu skaits 4-2=2.
$k = 3 \Rightarrow r_A = 5, r_B = 5$ Tā kā $r_A > M$, izeja no kolonnu cikla.

5) `sākuma_rinda=2, beigu_rinda=3`

v[0]	v[1]	v[2]	v[3]	v[4]
0	9	16	20	25

$k = 1 \Rightarrow r_A = 1, r_B = 2$ Derīgo variantu skaits 2-1=1.
$k = 2 \Rightarrow r_A = 2, r_B = 3$ Derīgo variantu skaits 3-2=1.
$k = 3 \Rightarrow r_A = 4, r_B = 5$ Derīgo variantu skaits 5-4=1.
$k = 4 \Rightarrow r_A = 5, r_B = 5$ Tā kā $r_A > M$, izeja no kolonnu cikla.

6) `sākuma_rinda=3, beigu_rinda=3`

v[0]	v[1]	v[2]	v[3]	v[4]
0	3	3	7	9

$k = 1 \Rightarrow r_A = 3, r_B = 5$ Derīgo variantu skaits 5-3=2.
$k = 2 \Rightarrow r_A = 5, r_B = 5$ Tā kā $r_A > M$, izeja no kolonnu cikla.

Aprakstītā algoritma izpildes laika sarežģītība ir $O(N^2M)$.

Ciparu aizvietošana

Risinājumam ir divas relatīvi neatkarīgas daļas - A) dažādo skaitļu ar doto pierakstu skaita atrašana un B) mazākā skaitļa ar šo pierakstu atrašana.

Vispirms atrisināsim uzdevumu A.

Lai to paveiktu, caurskatīsim simbolu virkni no beigām un katrā pozīcijā saglabāsim masīvu garumā 10, atzīmējot, vai šajā vietā var sākties katrs no desmit decimālajiem cipariem. Apstrāde no beigām ir izdevīga, jo var izmantot iepriekš apstrādāto informāciju.

Ja kārtējā pozīcijā ir 0, tad šajā pozīcijā var sākties tikai 0. Visu pārējo ciparu pierakstu apstrādei jau zinām, ka pirmais cipars ir 1. Piemēram, lai noteiktu, vai šajā pozīcijā var sākties decimālā cipara 7 pieraksts, pietiek pārbaudīt, ka nākamajā pozīcijā var sākties 3.

Uzdevuma tekstā dotajai virknei 100111 šo masīvu vērtības būtu šādas:

	1	0	0	1	1	1
0	nē	jā	jā	nē	nē	nē
1	jā	nē	nē	jā	jā	jā
2	jā	nē	nē	nē	nē	nē
3	nē	nē	nē	jā	jā	nē
4	jā	nē	nē	nē	nē	nē
5	nē	nē	nē	nē	nē	nē
6	nē	nē	nē	nē	nē	nē
7	nē	nē	nē	jā	nē	nē

8	nē	nē	nē	nē	nē	nē
9	jā	nē	nē	nē	nē	nē

Zinot šo informāciju, varam pakāpeniski aprēķināt dažādo variantu skaitu, atkal sākot apstrādi no rindas beigām. Ar s_i apzīmēsim dažādo skaitļu skaitu, kas sākas virknes pozīcijā $i(1 \leq i \leq G)$. Atļaujam skaitļiem virknes vidū sākties arī ar 0 (no dotā ir zināms, ka visas virknes pirmais simbols būs 1). Uzskatām, ka tukšo virkni varam izveidot vienā veidā ($s_{G+1} = 1,\ s_{G+2} = s_{G+3} = s_{G+4} = 0$).

Tālāk varam izmantot sakarības:

- $S_G = S_{G+1}$
- ja pozīcijā i var sākties 2 vai 3 pieraksts, tad $S_G = S_G + S_{G+2}$
- ja pozīcijā i var sākties 4, 5, 6 vai 7 pieraksts, tad $S_G = S_G + S_{G+3}$
- ja pozīcijā i var sākties 8 vai 9 pieraksts, tad $S_G = S_G + S_{G+4}$

Mūsu piemēra virknei dažādo skaitļu skaita vērtības mainītos šādi:

indekss	1	2	3	4	5	6	7	8	9	10
cipars	1	0	0	1	1	1				
s_i	14	4	4	4	2	1	1	0	0	0

Jāatceras, ka summa jāaprēķina pēc moduļa, tāpēc katrā solī vienkāršas saskaitīšanas vietā jāsaskaita un jāpaņem atlikums pēc norādītā moduļa.

Līdz ar to uzdevums A ir atrisināts.

Uzdevuma B atrisināšanai vispirms ievērosim, ka decimālam ciparam ar lielāku vērtību atbilst tikpat gara vai garāka bināro ciparu virkne. Tāpēc mazāko iespējamo skaitli varam meklēt no pieraksta beigām katru reizi ņemot lielāko iespējamo decimālo ciparu - to, kura binārais pieraksts ir visgarākais. Tas nodrošinās, ka skaitļa decimālajā pierakstā ir mazākais iespējamais ciparu skaits un pats skaitlis ir ar vismazāko vērtību.

Lai šo procesu varētu ērti organizēt, nepieciešams papildināt iepriekš aplūkoto masīva "vai šeit var sākties attiecīgā cipara pieraksts" aizpildīšanu ar viendimensiju masīva "kāda ir lielākā decimālā cipara vērtība, kas beidzas šajā pozīcijā, vērtība" aizpildīšanu.

To ir vienkārši realizēt - brīdī, kad zinām, ka pozīcijā i var sākties decimālā cipara c pieraksts, varam atzīmēt, ka lielākais cipars, kas beidzas pozīcijā $i + c\_garums - 1$ ir c. Ņemot vērā to, ka binārās virknes apstrāde noris no beigām uz sākumu, lielākā cipara vērtība noteiktā pozīcijā var tikai palielināties. Svarīgi, ka noteiktā pozīcijā ar vienu un to pašu garumu var beigties tikai viena decimālā cipara pieraksts un nav tādas pozīcijas, kurā nebeidzas neviens cipars.

Mūsu piemērā masīva "kāda ir lielākā decimālā cipara vērtība, kas beidzas šajā pozīcijā, vērtība" elementi būtu šādi:

indekss	1	2	3	4	5	6
binārais cipars	1	0	0	1	1	1
lielākais decimālais cipars, kas šeit beidzas	1	2	4	9	3	7

Decimālā skaitļa pēdējais cipars ir 7 (lielākais, kas beidzas 6. pozīcijā). Šis decimālais cipars aizņem trīs binārās pozīcijas, tāpēc kā nākamā jāaplūko 3. pozīcija. Tātad priekšpēdējais decimālais cipars ir 4. Esam "iztērējuši" visus bināros ciparus, tāpēc visi decimālā skaitļa cipari ir atrasti un esam ieguvuši, ka meklētā skaitļa pieraksts ir 47.

Algoritma sarežģītība ir $O(G)$. No realizācijas detaļām interesanti atzīmēt, ka masīva “vai šeit var sākties attiecīgā cipara pieraksts” realizācijai pietiek ar 5×10 elementiem, kuru vērtības cikliski tiek atjauninātas.

Nav apakšvirkne

Šis ir klasisks dinamiskās programmēšanas uzdevums.

Izveidosim $N \times G$ elementu masīvu P, kur $P[burts, i]$ elementa vērtība nozīmēs “virknes, kas sākas ar $burts$, un kam līdz dotās virknes X beigām var atrast visas iespējamās apakšvirknes, lielākais garums”.

Šo masīvu aizpildīsim no beigām un ievērosim, ka, ja $burts \neq X[i]$, tad $P[burts, i] = P[burts, i+1]$, bet, ja $burts = X[i]$, tad $P[burts, i] = 1 + \min\limits_{\text{burts}} P[burts, i+1]$.

Lai atvieglotu pirmās neatrodamās apakšvirknes vērtību, katrai i vērtībai ir vērts saglabāt mazāko $burts$ vērtību, kurai atbilstošā $P[burts, i]$ vērtība sakrīt ar mazāko vērtību starp visām $P[j, i]$.

Aplūkosim, kā izskatās masīva P aizpildījums uzdevuma tekstā dotajā piemērā:

	d	a	b	a	c	b	a	d	a	c
a	2	2	2	2	1	1	1	1	1	0
b	2	2	2	1	1	1	0	0	0	0
c	2	2	2	2	2	1	1	1	1	1
d	2	1	1	1	1	1	1	1	0	0
min	**2**	**1**	1	1	1	1	**0**	0	0	0
kurš	**a**	**d**	d	b	a	a	**b**	b	b	a

Aplūkojot vērtības, kas atbilst X pirmajam burtam, varam pārliecināties, ka virknē X var atrast visas apakšvirknes garumā 2, un pirmā neatrodamā virkne būs garumā trīs. Lai to atrastu, ir jāmeklē pēc kārtas pirmās kolonnas, kur mainās kolonnas minimuma vērtība un jāņem alfabēta pirmais atbilstošais burts, kas nosaka šo minimumu.

Aprakstītā risinājuma izpildes laika sarežģītība ir $O(NG)$.

Ja programmā tiek izmantota pāreja no burta uz tā kārtas numuru alfabētā, tad jāatceras, ka šajā uzdevumā tiek izmantots latviešu, nevis angļu alfabēts.

Peļķes

Tā kā plāksnes nedrīkst pārklāties, tad, lai kā arī tās būtu novietotas, būs iespējams (domās) novilkt vertikālu vai horizontālu līniju tā, ka viena no plāksnēm būs viena puse un otra plāksne otra puse.

Turpmāk pieņemsim, ka iedomātā dalījuma līnija būs vertikāla un atradīsim labāko atbildi šajā gadījumā.

Sakārtosim visas peļķu koordinātas (punktus) pēc x koordinātas (ja x koordinātas ir vienādas, y koordināšu secība ir patvaļīga).

Ciklā iesim cauri sakārtoto punktu masīvam un aprēķināsim, kāda būtu atbilde, ja šī peļķe atrastos pie kreisākās plāksnes labās malas.

Lai to aprēķinātu, mums vajag atrast minimālo plāksnes augstumu un platumu, lai pārklātu punktus pa kreisi no iedomātās līnijas un minimālo augstumu un platumu, lai pārklātu punktus pa labi no iedomātās līnijas.

To mēs varam aprēķināt laikā $O(peļķu\ skaits)$ – pieņemsim, ka mēs jau zinām mazāko taisnstūri, kāds nepieciešams, lai pārklātu pirmos i punktus (sakārtotajā peļķu masīvā), tad lai aprēķinātu mazāko taisnstūri kāds nepieciešams, lai pārklātu pirmos $i+1$ punktus ir nākusi klāt tikai viena peļķe un iepriekš zināmajam taisnstūrim varam noskaidrot, vai ir jāmaina kreisā, labā, augšējā vai apakšējā mala, lai pārklātu arī šo punktu un to var izdarīt laikā $O(1)$. Esam noskaidrojuši taisnstūra izmērus kas nepieciešams, lai pārklātus visus punktus kas atrodas pa kreisi no iedomātās līnijas. Līdzīgā veidā varam aprēķināt taisnstūra, kas nepieciešami lai pārklātu punktus pa labi no iedomātās līnijas tikai rēķinot no masīva beigām, izmērus.

Tad katrai iedomātajai līnijai, kurai tagad zinām minimālo kreiso un minimālo labējo taisnstūri, jānoskaidro minimālais kopējais taisnstūris, lai varētu pārklāt abas daļas – jāatceras, ka taisnstūri var būt arī pagriezti.

Risinājuma izpildes laika sarežģītība ir $O(NlogN)$.

Receptes

Vispirms jāielasa pieejamo izejvielu daudzumi un tad ciklā pēc kārtas jāapstrādā visas receptes. Tālāk iztirzāsim vienas receptes apstrādi.

Aplūkosim tikai tās izejvielas, kuras nepieciešamas receptē - t.i., kuru nepieciešamais daudzums nav 0. Lielāko iespējamo toršu skaitu nosaka tā sastāvdaļa, kura starp visām sastāvdaļām pietiek **mazākajam** toršu daudzumam. Tas nozīmē, ka katrai receptei jāatrod un jāizvada mazākā lieluma $\left[\frac{pieejams_i}{receptē_i}\right]$ vērtība, kur $pieejams_i$ ir pieejamais i-tās izejvielas daudzums, bet $receptē_i$ - receptē nepieciešamais šīs izejvielas daudzums.

Risinājuma izpildes laika sarežģītība ir $O(nk)$, kur n - dažādo izejvielu, bet k - dažādo recepšu skaits.