LATVIJAS 22. INFORMĀTIKAS OLIMPIĀDES II POSMA UZDEVUMU ĪSI ATRISINĀJUMI

1. "BANKA"

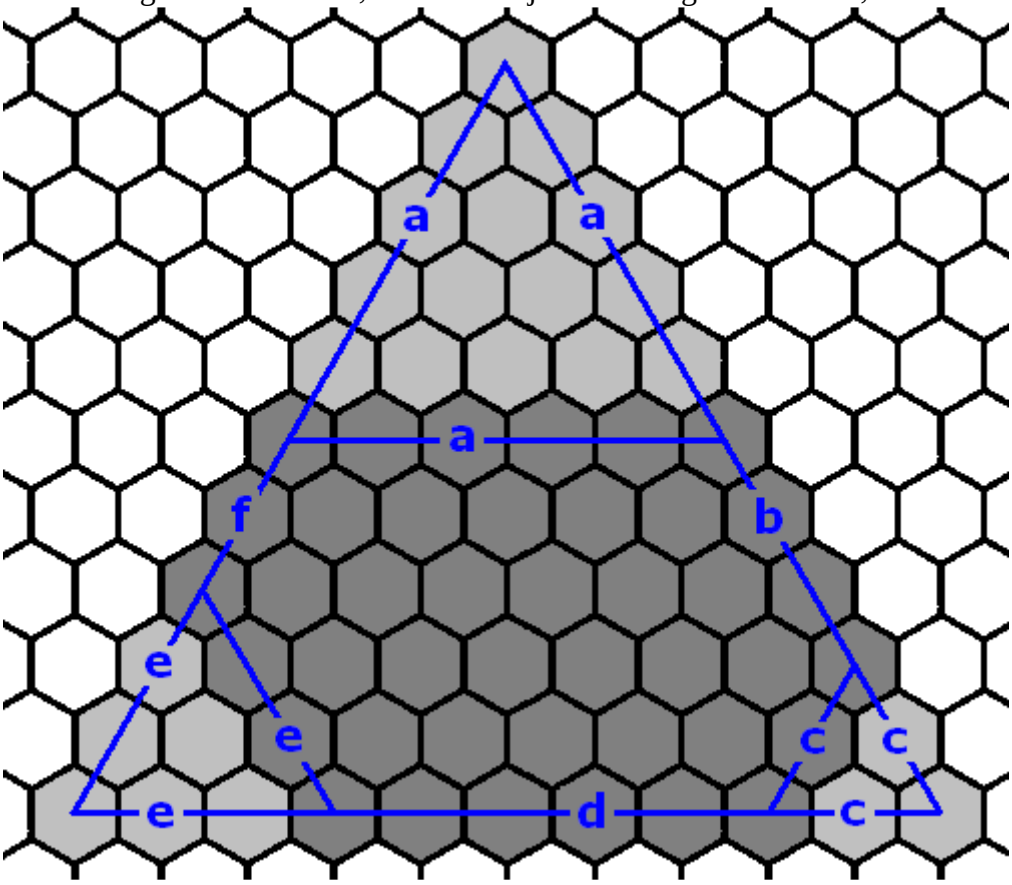
Pieņemsim, ka pēc dažām veiktajām operācijām klienta kontā ir $V \geq 0$ naudas vienības un kopš pēdējās dubultotās iemaksas veiktas I ($0 \leq I \leq N-K$) iemaksas. Sākumā V = S un I = 0. Ja kārtējā operācija ir naudas izņemšana, tad pietiek attiecīgi samazināt V vērtību, bet, ja tā ir naudas iemaksa, tad:

1) vispirms palielinām I;
2) ja tagad I=N−K+1, tad dubultojam šo iemaksu, attiecīgi palielinot V vērtību, un uzstādām I vienādu ar 0.

Beigās izvadām V vērtību.

2. "ŠŪNAS"

Ar figūras *laukumu* sapratīsim šūnu skaitu, ko šī figūra ietver. Apzīmēsim dotā sešstūra malu garumus ar *a*, *b*, *c*, *d*, *e*, *f* tā, kā redzams attēlā, un pagarināsim malas *b*, *d*, *f*, līdz to krustpunktam tā, ka izveidojas trijstūris. Redzam, ka dotā sešstūra laukums ir vienāds ar lielā trijstūra laukumu, no kura atņemti trīs mazo trijstūru laukumi. Lielā trijstūra malas garums ir *a+b+c*, bet mazo trijstūru malu garumi ir *a−1*, *c−1* un *e−1*.



Trijstūra ar malas garumu v laukums ir viegli izrēķināms: varam saskaitīt, cik šūnu tam ir katrā no horizontālēm un iegūsim 1 + 2 + 3 + ... + v + (v+1) = = (v+1) * (v+2) / 2 (pēc aritmētiskās progresijas summas formulas). Šo summu pie dotajiem ierobežojumiem var aprēķināt arī ar vienkāršu ciklu, neizmantojot formulu.

3. "PASJANSS"

Risināsim nedaudz vispārīgāku uzdevumu, kurā uz kartītēm uzrakstītie skaitļi ne obligāti sākas ar 1. Ieviesīsim funkciju f(n, k, s), kas atrod uz k-tās kartītes uzrakstīto skaitli, ja sākumā ir n kartītes, uz kurām uzrakstīti skaitļi no s līdz s+n−1. Tad atbilde būs f(N, K, 1).

Ievērosim, ka sākumā un arī pēc katra gājiena izpildes katrā čupiņā kāršu skaits ir vienāds ar 3^g, kur g – veikto gājienu skaits (izņemot pēdējo čupiņu, kurā kāršu skaits var būt mazāks).

Mēģināsim atrast f(n, k, s) vērtību. Atradīsim lielāko a, ka skaitlis 3^a ir mazāks par kartīšu skaitu n. Ir iespējami divi gadījumi:

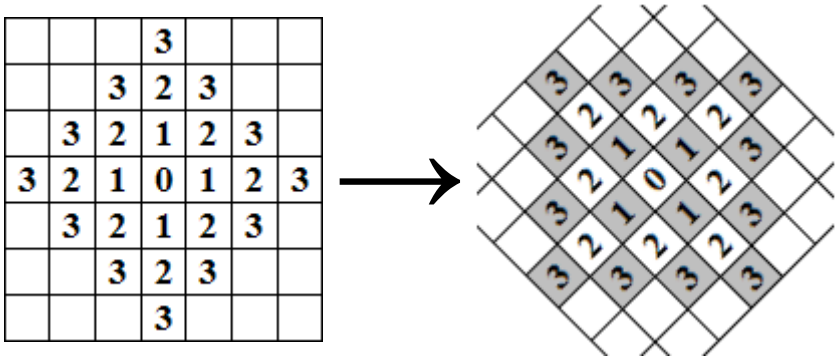
1) $3^a < n \leq 2 * 3^a$. Tas nozīmē, ka pirms pēdējā gājiena būs divas kartīšu čupiņas – pirmā ar 3^a kartītēm un otrā, kurā ir vismaz viena un ne vairāk par 3^a kartītēm.
    a. Ja $k \leq 3^a$, tad meklētā kartīte atrodas pirmajā no šīm čupiņām, tātad, jāatrod $f(3^a, k, s)$ vērtība.
    b. Ja $k > 3^a$, tad meklētā kārts atrodas otrajā no šīm čupiņām un jāatrod $f(n - 3^a, k - 3^a, s + 3^a)$ vērtība.
2) $2 * 3^a < n \leq 3 * 3^a$. Tad $n = 3^a + 3^a + t$, kur $1 \leq t \leq 3^a$. Tas nozīmē, ka pirms pēdējā gājiena būs trīs kartīšu čupiņas – pirmās divas ar 3^a kartītēm un trešā ar t kartītēm.
    a. Ja $k \leq t$, tad meklētā kārts atrodas trešajā no šīm čupiņām, tātad, jāatrod $f(t, k, s + 3^a + 3^a)$ vērtība.
    b. Ja $t < k \leq t + 3^a$, tad meklētā kārts atrodas pirmajā no šīm čupiņām un jāatrod $f(3^a, k - t, s)$ vērtība.
    c. Ja $t + 3^a < k \leq t + 3^a + 3^a$, tad meklētā kārts atrodas otrajā no šīm čupiņām un jāatrod $f(3^a, k - t - 3^a, s + 3^a)$ vērtība.

Lai atrastu jauno f vērtību, atkal pielietojam aprakstīto soli. Katrā reizē kartīšu skaits samazinās; patiesībā var pierādīt, ka veicamo soļu skaits sakrīt ar gājienu skaitu, jo mēs katrā reizē it kā paņemam vienu gājienu atpakaļ.

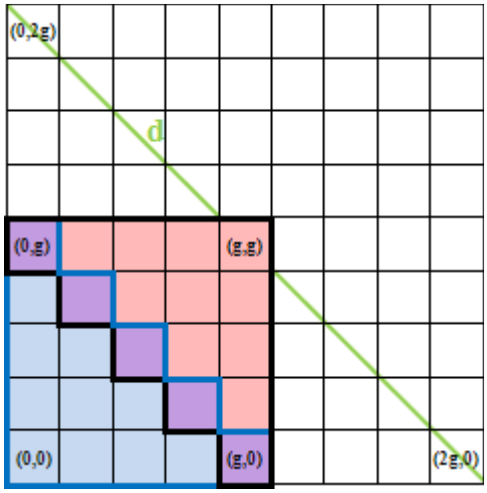
Aplūkosim piemēru: N = 70, K = 6.
Meklējam f(70, 6, 1). $a = 3$, $70 > 2 * 3^3$, tāpēc ir spēkā gadījums 2).
$t = 16$. $k \leq t$ (apakšpunkts a).
Tālāk meklējam f(16, 6, 55). $a = 2$, $16 \leq 2 * 3^2$, tāpēc ir spēkā gadījums 1).
$k \leq 3^2$ (apakšpunkts a).
Tālāk meklējam f(9, 6, 55). $a = 1$, $9 > 2 * 3^1$, tāpēc ir spēkā gadījums 2).
$t = 3$, $t < k \leq t + 3^1$ (apakšpunkts b).
Tālāk meklējam f(3, 3, 55). $a = 0$, $3 > 2 * 3^0$, tāpēc ir spēkā gadījums 2).
$t = 1$, $t + 3^0 < k \leq t + 3^0 + 3^0$ (apakšpunkts c).
Tālāk meklējam f(1, 1, 56). f(1, 1, s) = s, tāpēc atbilde ir 56.

4. "VĪRUSI"

Tā kā mūs interesē tikai doto divu rūtiņu savstarpējais novietojums, tad varam novietot vienu rūtiņu punktā (0,0), bet otru rūtiņu (x,y) tā, ka tās abas koordinātas ir nenegatīvas. Ja vienā gājienā drīkstam pārvietoties uz blakusesošu rūtiņu, tad no rūtiņas (0,0) līdz rūtiņai (x,y) varam aiziet x+y gājienos. Tā kā vīrusi ir divi, tad nepieciešamo gājienu (jeb sekunžu) skaits būs uz pusi mazāks. Precīzāk, vīrusi "satiksies", t.i., abi inficēs kādu rūtiņu, pēc $g := \left\lceil \frac{x+y}{2} \right\rceil$ gājieniem ($\frac{x+y}{2}$ vērtība noapaļota uz augšu līdz veselam skaitlim). Katrs no vīrusiem pēc g gājieniem būs inficējis $(g+1)^2+g^2$ rūtiņas (to var redzēt, iekrāsojot rūtiņas šaha galdiņa veidā un paskatoties uz tām 45 grādu leņķī, skat. attēlu). Atliek tikai atņemt kopīgo rūtiņu skaitu, tāpēc mēģināsim noteikt, cik rūtiņas inficēs abi vīrusi.



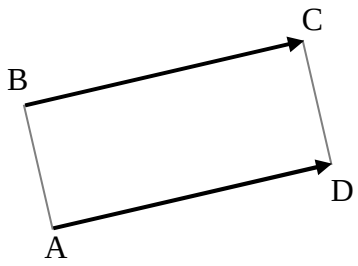
Iespējami divi gadījumi.



1) x+y dalās ar 2. Tad x+y = 2g, t.i., otrā rūtiņa atrodas uz diagonāles d (skat. attēlu). Ja x = y = g, tad ar abiem vīrusiem inficētās rūtiņas ir (0,g), (1,g-1), ..., (g-1,1), (g,0) – pavisam g rūtiņas. Attālinot otrā vīrusa sākuma rūtiņu no (g,g) par kādu skaitu vienību, kopīgo rūtiņu skaits samazinās par tikpat. Nav grūti redzēt, ka kopīgo rūtiņu skaits vienāds ar mazāko no x un y vērtībām.
2) x+y nedalās ar 2. Tad x+y = 2g-1. Bīdot otrā vīrusa sākuma rūtiņu pa šo diagonāli, kopīgo rūtiņu skaits mainās pa divām rūtiņām. Vingrinājums lasītājam – pārliecināties, ka kopīgo rūtiņu skaits izsakāms ar šādu formulu: 2 * (min(x, y) + 1).

Piezīme: rakstot risinājumu, jāņem vērā, ka atbilde var pārsniegt maksimālo int (longint) vērtību.

5. "TAISNSTŪRA VIRSOTNE"



Atrodam, starp kurām divām virsotnēm ir vislielākais attālums; apzīmēsim tās ar A un C, trešo doto virsotni – ar B, bet meklējamo virsotni – ar D (skat. attēlu). Tad ABCD ir taisnstūris (un, protams, arī paralelograms). Paralelograma pretējo malu veidotie vektori ir vienādi. Mūsu gadījumā vektori $\overrightarrow{BC}$ un $\overrightarrow{AD}$ ir vienādi, t.i., $C_x - B_x = D_x - A_x$ un $C_y - B_y = D_y - A_y$, no kurienes iegūstam $D_x = A_x - B_x + C_x$ un $D_y = A_y - B_y + C_y$.

Piezīme: lai atrastu vislielāko attālumu starp virsotnēm, pietiek salīdzināt attālumu kvadrātus (jo lielāks attāluma kvadrāts, jo lielāks attālums). Starp punktiem P un Q attāluma kvadrāts vienāds ar $(P_x - Q_x)^2 + (P_y - Q_y)^2$.

6. "INFORMĀTIKAS OLIMPIĀDES"

Šo uzdevumu var atrisināt ar dinamiskās programmēšanas metodi.

Ja fragmentu LIO, BOI, IOI vietā būtu tādi fragmenti, kas "nepārklājas", piemēram, ABC, DEF, GBB, tad uzdevumu varētu viegli atrisināt, izmantojot dinamiskās programmēšanas pieeju šādā veidā. Apzīmēsim doto virkni ar A, dotās virknes pirmo k simbolu veidoto apakšvirkni ar A_k, bet lielāko iespējamo šīs virknes vērtību ar M_k. Tad, ja mums ir zināma M_{i-3} vērtība kādam i, tad varam viegli izrēķināt M_i vērtību, mēģinot ievietot virknes (i-2)-ajā, (i-1)-ajā un i-tajā pozīcijā doto fragmentu. Mūsu gadījumā sanāktu:

```
papildus_punkti := 0;
if (var_ievietot(i-2, "ABC")) then
     papildus_punkti := max(papildus_punkti, X);
if (var_ievietot(i-2, "DEF")) then
     papildus_punkti := max(papildus_punkti, Y);
if (var_ievietot(i-2, "GBB")) then
     papildus_punkti := max(papildus_punkti, Z);
M[i] := max(M[i-1], M[i-3] + papildus_punkti);

function var_ievietot(sakums, fragments)
begin
  if (A[sakums] = '?' or A[sakums] = fragmenta pirmais burts) and
  (A[sakums+1] = '?' or A[sakums+1] = fragmenta otrais burts) and
  (A[sakums+2] = '?' or A[sakums+2] = fragmenta trešais burts) then
       var_ievietot := true
  else
       var_ievietot := false;
end;
```

Skaidrs, ka $M_1 = M_2 = 0$. Inicializācijā varam izrēķināt M_3 un sākt ar i = 4 (t.i., meklēt M_4 vērtību), bet vieglāk ir pieņemt, ka $M_0 = 0$ un sākt ar i = 3 (t.i., meklēt M_3 vērtību).

Bet mūsu gadījumā dotie fragmenti ir *viltīgi* – tie var pārklāties (piemēram, virknēs LIOI, BOIOI, IOIOI dotie fragmenti pārklājas). Tāpēc ir vērts saglabāt lielāko iespējamo apakšvirknes A_k vērtību, ja tā beidzas ar kādu no dotajiem fragmentiem LIO, BOI, IOI. Šīs vērtības apzīmēsim ar LIO_k, BOI_k, IOI_k.

Fragmentiem LIO un BOI šīs vērtības ir viegli izrēķināmas, jo šie fragmenti nevar pārklāties ar kādu pirms tam esošu fragmentu (-1 nozīmē, ka virkne nevar beigties ar attiecīgo fragmentu):

```
if (var_ievietot(i-2, "LIO")) then LIO[i] := M[i-3]+X else LIO[i] := -1;
if (var_ievietot(i-2, "BOI")) then BOI[i] := M[i-3]+Y else BOI[i] := -1;
```

Fragments IOI turpretī var no kreisās puses pārklāties gan ar fragmentu LIO, gan ar BOI, gan ar IOI (protams, jāaplūko arī iespēja, kad tas nepārklājas ar citiem fragmentiem):

```
if (var_ievietot(i-2, "IOI")) then
     IOI[i] := max(LIO[i-1], BOI[i-2], IOI[i-2], M[i-3]) + Z
else
     IOI[i] := -1;
```

M_i vērtība izrēķināma kā lielākā no LIO_i, BOI_i, IOI_i un M_{i-1} vērtībām.

Aplūkosim masīvu aizpildījumu piemēram X=1, Y=7, Z=3 ar doto virkni BO?IO?OI??Z.

i	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
A		B	O	?	I	O	?	O	I	?	?	Z
LIO	0	0	0	-1	-1	1	-1	-1	-1	-1	-1	-1
BOI	0	0	0	7	-1	-1	-1	-1	14	-1	-1	-1
IOI	0	0	0	-1	-1	-1	10	-1	13	-1	17	-1
M	0	0	0	7	7	7	10	10	14	14	17	17