SUURIN YKKÖSTEN NELIÖ

1
1
1
1
1
1

Tästä ei saa kertyä pisteitä

1 1 1 1 1 1 1 tästä ei myöskään

```
1 1 → 1 2 → 1 2 → 1 2
1 1   2 1   2 2   2 3
```

Jos oikeella & alla on ykkönen+, korota niitä yhellä

```
1 1 1   1 2 1   1 2 2   1 2 2   1 2 2   1 2 2   1 2 2
1 1 1 → 2 1 1 → 2 2 1 → 2 2 2 → 2 3 2 → 2 3 3 → 2 3 3
1 1 1   1 1 1   1 1 1   1 1 1   2 1 1   2 2 1   2 2 2
```

Jos vasemmalla & alla on ykkönen+, korota alla olevaa

~~Jos oikeella & alla on 1+, korota~~

Jos yllä & oikeella on 1+, korota oikealla olevaa

```
1 2 2   1 2 2
2 3 3 → 2 3 3
2 3 2   2 3 3
```

MUTTA! Tästä ei nyt nää missä on se isoin kulma!

Mitä jos plussaa seuraavaan edellisen luvun?

```
1 1 1   1 2 1   1 2 3   1 2 3   1 2 3   1 2 3   1 2 3
1 1 1 → 2 1 1 → 2 3 1 → 2 3 4 → 2 5 4 → 2 5 9 → 2 5 9
1 1 1   1 1 1   1 1 1   1 1 1   3 1 1   3 6 1   3 6 10
```

```
1 2 3    1 2 3
2 5 9  → 2 5 9
3 9 10   3 9 19
```

eli yhen rivit asetetaan sillä, että tarkistetaan 2 vierestä ruutua. Mut entä jos 2:n levee?

```
1 1   1 2   1 2   1 2   1 2   1 2   1 2   1 2
1 1 → 2 1 → 2 3 → 2 5 → 2 5 → 2 5 → 2 5 → 2 5
1 1   1 1   1 1   3 1   3 6   3 9   3 9   3 9
1 1   1 1   1 1   1 1   1 1   4 1   4 10  4 14
                                          5 20
                                          6 27
```

Tää ei oo neliö!

Voinko tarkistaa ruutumaisuuden isointa nro:a lukujonoon, joka tulee neliöitten kulmasta?

1	2	3	4	5	6	7
2	5	9	14	20	27	35
3	9	19	34	55	83	119
4	14	34	69	125	209	329
5	20	55	125	251	461	791
6	27	83	209	461	923	1715
7	35	119	329	791	1715	3431

Eli selvästi, neliöiden kulmaluvut on uniikkeja.

⇒ Aloita tarkastaan isoimmasta luvusta, ja eka joka on neliön kulmassa antaa isoimman ykkösten ~~ja~~ neliön oikean alakulman!

näistä ei tuu mitään koska vaan 1 1+ vieressä

```
         0 0 0 0
0 1 1 0    0 1 2 3      Mutta entäs erikoistapaukset?
0 1 0 0 →  0 2 0 0
           0 0 0 0
```

```
  1     1         1       1               1  1   1                 Problem!!
1 1 1 1 1 → 1 2 1 1 1 1 → Lopussa: → 1 2    1 2 3 1   Mitä jos
1   1       2     1                  2       3 1 1     just tästä
                                               1 1     kulmasta
                                                       alkaa
                                                       ~~matriisi?~~
                                                       neliö
```

Tuon takia,
```
1       1 1
1 1  ja   1
```
kulmat voi korottaa vain jos vielä joku ehto täyttyy...

```
1 1 1     1 2 1     1 2 3     1 2 3     1 2 3     1 2 3
1 1 1  →  2 1 1  →  2 3 1  →  2 3 4  →  2 5 4  →  2 5 9
  1 1       1 1       1 1       1 1       1 1       6 1
```

⇒ Menis numerot sekaan ko sen aloituskulma on yli 1

```
  1 2 3     1 2 3
→ 2 5 9  →  2 5 9
  6 10      6 16  ← ei osu neliö-lukujonoon!
```
9 & 6 ei osu myöskään. 5 osuu! ⇒ Suurin 1:ten neliö on 2×2

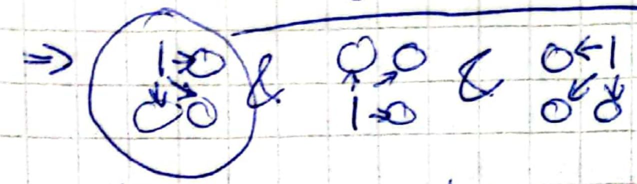
Ei voi myöskään laittaa vinottain koska laskee väärin:

```
1 1 1     1 1 1     1 1 1
1 1 1  →  1 2 1  →  1 2 1  → Mutta eipä oo kokonainen neliö!
  1 1       1 1       1 3
```

```
1 1       1 1       1 1       1 1
  1 1  →  2 1    →  2 3   →   2 3
    1 1     1 1  →    1 1  →  4 5  ← tää ois väärin!!
      1       1         1     6
```

Tuosta tuli mieleen, että jos sivuilla olevat numerot kasvaa tarpeeks isoiksi, ne pitää sisällään neliön kulmalukuja!

Mut mitä jos tarkistanki 3 eri suuntaa?



Ja tää siis tarkistetaan aina ekana!!

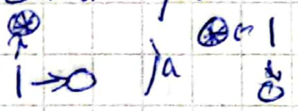
Sillo ei vois tulla mitään

1	1
2	3
4	5
	6

-tyylisiä

Mutta tulis lisää ~~ta~~rkistuksia ⇒ vie enemmän aikaa.

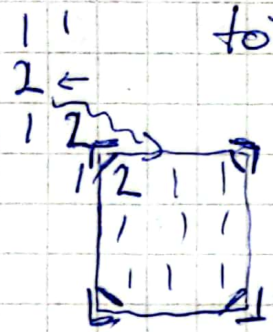
Entä jos



, mutta *-kohissa pitää olla >1?

Toki 1 1 / 2 ← / 1 2 toimis sitte samalai ku ylhäällä..

Melkeen, koska * ei oo >1.



Mut taas, jos tuosta alkais neliö nii ois eri luvut.

Mutta tässäpä on mielenkiintonen ongelma:

1	1		
1	1	1	1
	1	1	1
	1	1	1

→

1	2		
2	1	1	1
	1	1	1
	1	1	1

→

1	2		
2	3	1	1
	1	1	1
	1	1	1

→

1	2		
2	5	1	1
	1	1	1
	1	1	1

→

1	2		
2	5	6	1
	6	1	1
	1	1	1

→

1	2		
2	5	6	7
	6	7	1
	1	1	1

→

1	2		
2	5	6	7
	6	7	8
	1	1	1

→

1	2		
2	5	6	7
	6	13	8
	7	1	1

→

1	2		
2	5	6	7
	8	13	21
	7	14	1

→

1	2		
2	5	6	7
	8	13	21
	7	14	22

→

1	2		
2	5	6	7
	8	13	21
	7	21	22

→

1	2		
2	5	6	7
	8	13	21
	7	21	43

⇒ 13 ja 43 ei kuulu neliökulmien joukkoon, vaikka tässä meiän pitäs saaha 3x3 vastaukseks!

Entä jos tarkistaa onko nuo nuolten osoittamat ruudut tyhjiä? Jos ne on, jos ympyrän sisällä >1 ja jos 6:n ja 8:n kohalla on 1 nii nollaa tuo vitonen? MUTTA! entä jos tu oiski lopussa ja jos se on neliön kulmaluku!

$$\begin{array}{cccc} 1 & 1 & 1 & \\ 1 & 1 & 1 & \\ 1 & 1 & 1 & 1 \\ & 1 & 1 & \end{array}$$

⇒ Alä nollaa vaan ala muuten vaan laskeen alusta seuraavia? Eli:

$$\begin{array}{cccc} 1 & 2 & 3 & \\ 2 & 5 & 9 & \\ 3 & 9 & 19 & 2 \\ & & 2 & 5 \end{array}$$

⇒ suurin on 3x3

Entäpä:

$$\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & \end{array} \rightarrow \begin{array}{cccc} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 1 & 1 & 1 & \end{array}$$

$$\rightarrow \begin{array}{cccc} 1 & 2 & 3 & 4 \\ 2 & 5 & 4 & 5 \\ & 1 & 1 & 1 \end{array} \rightarrow \begin{array}{cccc} 1 & 2 & 3 & 4 \\ 2 & 5 & 9 & 5 \\ & 6 & 1 & 1 \end{array} \rightarrow \begin{array}{cccc} 1 & 2 & 3 & 4 \\ 2 & 5 & 9 & 14 \\ & 6 & 10 & 1 \end{array} \rightarrow \begin{array}{cccc} 1 & 2 & 3 & 4 \\ 2 & 5 & 9 & 14 \\ & 6 & 10 & 15 \end{array} \rightarrow \begin{array}{cccc} 1 & 2 & 3 & 4 \\ 2 & 5 & 9 & 14 \\ & 6 & 16 & 15 \end{array}$$

$$\rightarrow \begin{array}{cccc} 1 & 2 & 3 & 4 \\ 2 & 5 & 9 & 14 \\ & 6 & 16 & 21 \end{array}$$

tän pitäs antaa vastaukseks 3x3

$\boxed{\text{Tähän väliin vähän koodausta!}}$

- Nämähän meneenyt ihan päin hemmettiä xD
  Nään 3 vaihtoehtoa:

1.) Ei käytä tuota neliökulma lukujonoa vertauksiin
2.) Erittele eri (matriisi)neliöt toisistaan (päällekkäiset siis)
3.) Tarkistaa sittenki lopuksi joka neliön yhtenäisyyden

Tuli mieleen sellanen idea, että entä jos päivittämisen jälkeen palautan vanhan luvun ennalleen? Silloin tarpeeks kaukana olevat neliöt ei häirittis toisiaan... Eli

$$\begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \rightarrow \begin{array}{ccc} 1 & 2 & 1 \\ 2 & 1 & 1 \\ 1 & 1 & 1 \end{array} \rightarrow \begin{array}{ccc} 1 & 1 & 3 \\ 2 & 3 & 1 \\ 1 & 1 & 1 \end{array} \rightarrow \begin{array}{ccc} 1 & 1 & \emptyset \\ 2 & 3 & 4 \\ 1 & 1 & 1 \end{array} \rightarrow \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 5 & 4 \\ 3 & 1 & 1 \end{array} \rightarrow \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 9 \\ 3 & 6 & 1 \end{array} \ldots$$

Tällöin mä tarvisin 2 riviä kerrala

⇒ Testasin, ei näytä olevan merkittävää vaikutusta (tai siis ei ratkase kaikkia ongelmia).

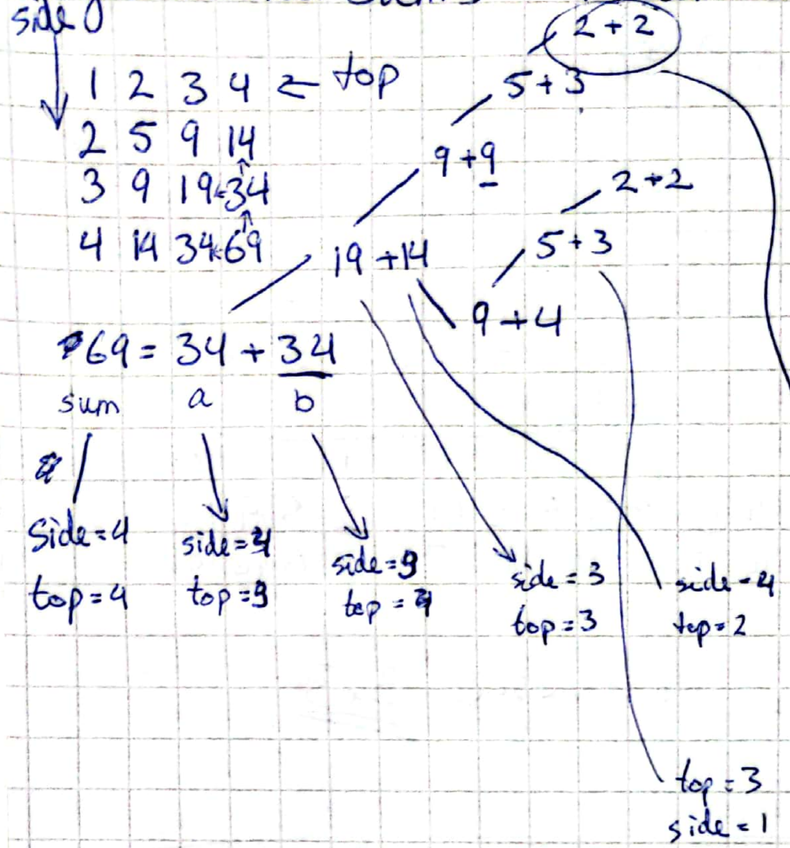
MITÄ jos vedän saadun kulmaluvun alaspäin ensimmäiseen vastaan tulevaan neliökulmalukuun ja käyn läpi sen matriisikoon (tarkastan siis)?

→ Ei tarvikaan, jos molemmat ∅ on tyhjiä!

Scanned with CamScanner

getCornerSums rekursio. n=~~#~~4

(huom! jokaisessa on myös se +1)

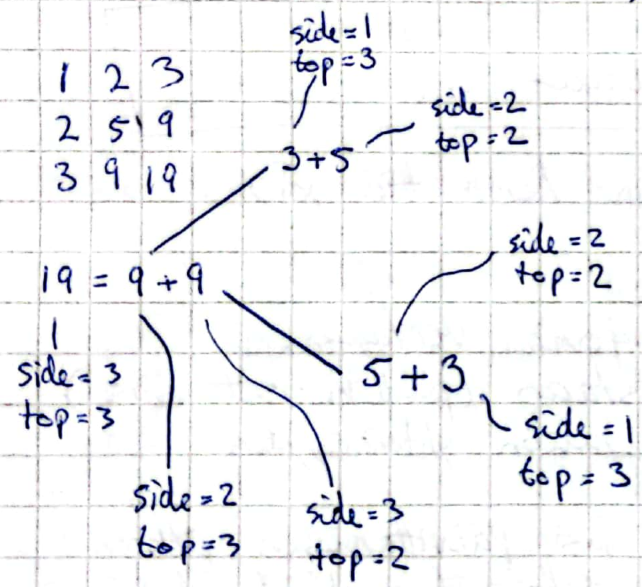



```
if (a=b) {
list.push(a+b)
}
```

if side=2 & top=2
return 5

a:ssa top − 1
b:ssa side − 1

if side=1
return top
if top=1
return side



Iso suorituskyvyn syöjä on tää rekursio. Esim. jos on 9x12 matriisi, nii rekursio antaa luvut

n= 2	3	4	5	6	7	8	9
5	19	69	251	923	3431	12869	48619

ja tekee prosessin aikana 25739 rekursiokutsua..

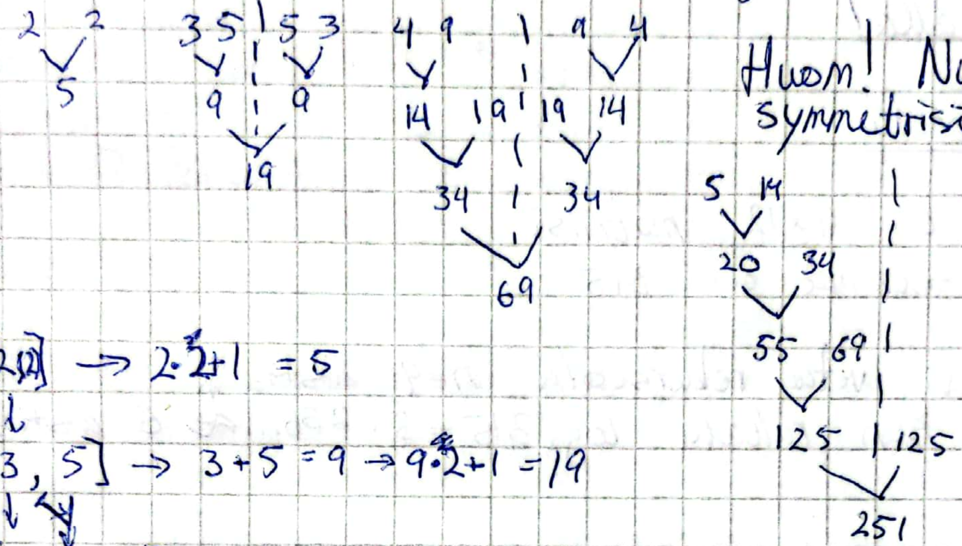
sillä välin ku muu ohjelma teki vaan 282 looppia.

Max kulmasumma oli 315. → Voiko parantaa?

Vaihtoehdot:
1. Tehokkaampi rekursio
2. Yritä hyödyntää maks. kulmasummaa maksimi neliökoon arvioimiseen - koska maks neliökoko kyseisellä summalla selviää vasta rekursion jälkeen!!

Tai 1 & 2 tietty :D

① Eli mitä jos tällä kerralla hyödynnän saatuja tuloksia?



Huom! Nuo on symmetrisiä!

[2,2] → $2 \cdot 2 + 1 = 5$
+1 ↓
[3, 5] → 3+5 = 9 → $9 \cdot 2 + 1 = 19$
+1 ↓
[4, 9, 19] → 4+9 = 14 → 14+19 = 34 → $34 \cdot 2 + 1 = 69$
+1 ↓
[5, 14, 34, 69]
+1 ↓
[6, 20, 55, 125, 251]

Huom! Koska tuo suurin summa vaihtuu aina, se pitää tallentaa lisäksi toiseen listaan!

② Oisko joku lukujono/sarja, joka antais aina ylemmän rajan?

	+1 jokaseen	2	3	4	5	6	7	8
Esim. 5^n ? ⇒	5,	25,	125,	625,	3125	15625,	78125,	390625
VS	5,	19,	69,	251,	923,	3431,	12869,	48619
ero:	0	6	56	374,	2202,	12194,	65256,	342006

⟹ Nousee monotonisesti :)
(kasvaa)

Eli n-suuruisen neliön kulmasumma ei voi koskaan olla suurempi kuin 5^n

$\sqrt[5]{315} = 3,15982\ldots$ ← ok, tää ei selvästikään tee mitään xD

Hmm, tarvin varmaan sittenki alarajan:

4^n? 5, 19, 69, 251, 923, 3431, 12869, 48619
↳ ~~4, 16, 64, 256, 1024, 4096~~ → 4^n ei toimi alarajana

3^n? 3, 9, 27, 81, 243, 729, 2187, 6561 ⇒ Ero kasvaa monotonisesti!
ERO:

Max
$n =$ 2 3 4 5 6 7 ...
cornersum = 5, 19, 69, 251, 923, 3431 ...
$\log_3$(cornersum) = 1.46, 2.7, 3.9, 5.1, 6.2, 7.4 ...

YLÖSPÄIN

⇒ Eli, jos pyöristän $\log_3$(kulmasumma), niin se antaa mulle luvun, jota suurempi neliön sivun pituus ei voi varmasti olla kyseisellä kulmasummalla!

Minkä ansiosta:

- Kun mulla on 9×12 matriisi
- ~~Kun~~ Max kulmasumma on 315

⇒ Mun ei tarvi vetää rekursiota $n=9$ asti
⇒ Voin antaa sen rullata $\log_3 315 = 5.236... \approx 6$ asti!

Now (both) improvements are implemented:
Original: 25739 recursive calls, 0 loops in total
New: 4 recursive calls, 6 loops in total
} in this specific function

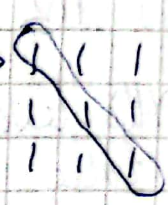
Oho, lipsahti vähä enkkua väliin :DDD

- Periaatteessa nyt on sellanen "ongelma, että matriisi voi olla vaikkapa 9×15, mutta suurin kulmasumma 92377
⇒ $\log_3 92377 \approx 10{,}407... \approx \underline{11 = n}$ ⇒ Turhan korkea n, koska eihän matriisissa voi olla niin isoa neliötä...
- Siinähän voi tulla myös out-of-bounds virhe, jos matriisi on esim 3×60, ja alkaa isolla n tarkastaan neliötä...

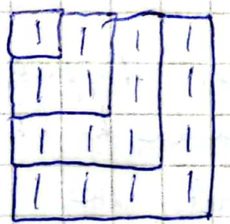
⇒ Korjattu!!

So, 2 days ago I talked about this w/ my friend (Bruce), (oho, erkku xD) ja se sano että on varmaan parempiki dynamic programming tapa tähän.

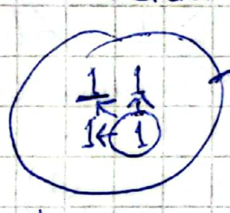
→ Se sai mut miettimään lisää:



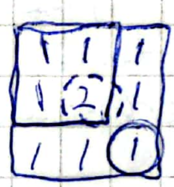
- Jokaisella neliöllä on uniikki diagonaali. →
  - Se ei mee ristiin minkää muun neliön kans. ELI: jos tallennan vastaukset (/tulokset) pelkästään siihen, nii ne ei mee päällekäin minkään muun neliön kans
- Sit voisin tarkistaa <u>heti</u>, onko kyseessä neliö:



⇒ Menee taas rivi kerrallaan, mutta tällä kertaa "työntämisen" sijasta voisin "vetää" tuloksen:



— jos nuolten osoittamissa suunnissa on ≥1, niin plussaa ympyrään vasemmassa yläkulmassa oleva luku.



⇒ Nyt 2x2 neliö on tarkistettu, ja se on yhtenäinen. Kun päästään alempaan ympyrään, niin pitää tarkistaa ~~oikean~~ → [vasemman yläkulman (2) numeron verran] elementtejä ylös ja vasemmalle. Jos ne on <u>kaikki</u> ≥1, niin neliö on yhtenäinen, ja sen koko on nyt <u>3x3</u>. Samoin, ympyrän luvusta tulee 1+2=<u>3</u>

Huom! Tällä metodilla ruudun numeroa voi nostaa vaan yhesti! Ja koska jokaisella neliöllä on oma diagonaali, niin tulokset ei sekoitu.

$$\begin{matrix}1&1&1\\1&1&1\\1&1&1\end{matrix} \rightarrow \begin{matrix}1&1&1\\1&1&1\\1&1&1\end{matrix} \rightarrow \begin{matrix}1&1&1\\1&1&1\\1&1&1\end{matrix} \rightarrow \begin{matrix}1&1&1\\1&1&1\\1&1&1\end{matrix} \rightarrow \begin{matrix}1&1&1\\1&2&1\\1&1&1\end{matrix} \rightarrow \begin{matrix}1&1&1\\1&2&2\\1&1&1\end{matrix} \rightarrow \begin{matrix}1&1&1\\1&2&2\\1&1&1\end{matrix}$$

$$\rightarrow \begin{matrix}1&1&1\\1&2&2\\1&2&1\end{matrix} \rightarrow \begin{matrix}1&1&1\\1&2&2\\1&2&3\end{matrix}$$

<u>NYT</u> näistä luvuista näkee suoraan, kuinka ison neliön alakulmassa ne on.

1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1
1	1	1	→	1	1	1	→	1	1	1	→	1	1	1	→	1	2	1	→	1	2	2	→	1	2	2
0	1	1		1	1	1		1	1	1		0	1	1		0	1	1		0	1	1		0	1	1

1	1	1
1	2	2
0	1	2

⇒ Huom! Tässä pitää miettiä mitä tapahtuu, jos tarkistettavat sivut (ylös, vasen) ei ookaan kaikki ≥1 ...

Ylläolevan perusteella näyttäis siltä, että ympyrään tulee se luku, kuinka kaukana lähin ← tai ↑ suunnassa oleva nolla on sitä. Yllä $n=2$.

1	1	1	1		1	1	1	1
1	2	1	1		1	2	1	1
1	1	3	1		1	1	3	0
0	1	1	3		1	1	1	1

Näyttäis pätevän!!!

1	1	1	1	1	1		1	1	1	1	1	1
1	1	1	1	1	0		1	2	2	2	2	0
1	1	0	1	1	1	⇒	1	2	0	1	2	1
1	1	1	1	1	1		1	2	1	1	2	2
1	0	1	1	1	1		1	0	1	2	2	3
1	1	1	1	1	1		1	1	1	2	3	3

tän se vaan skippaa koska ⇄ ei oo kaikki ≥1 AHAA, no siinähän tapauksessa noita missä on pikkunuoli (↕) ei tarvi myöskään tarkistaa!

~~Lopuksi käy vä~~ Samalla ku käy noita läpi, nii tallenna suurin tähän mennessä löydetty neliö, jonka palautat lopuksi. Testausta oudolla muodolla:

		1	1	1		1				1	1	1		1
		1	1	1	1					1	2	2	1	
1	1	1	0	1	1	1		1	1	1	0	1	2	1
1	1	1		1	1	1	⇒	1	2	2		1	2	2
1	1	1	1		1	1		1	2	3	1		1	2
	1	1	1		1	1			1	2	2		1	2
					1	1							1	2
					1	1							1	2

(tyhjät = 0)

Näyttäis kyl toimivan

entä jos:

	0
2	1
3	1

⇒ mahtuu vaan 2x2 neliö

0	1	1
1	1	2
1	2	1

⇒ Mahtuu vaan 2x2 neliö

eli

a	b
c	d

$d = 1 + \min(a, b, c)$, koska se kertoo maksimi neliön koon, joka voi kyseiseen ruutuun tulla

Scanned with CamScanner

TULOKSET:

LOPULLINEN: (kaikkine alempine parannuksineen)

20a looppia

25 riviä koodia

20x12 matriisi

Vanha algoritmi (semi-dynaaminen):

- 4019 looppia, 10 rekursiokutsua

Uusi algoritmi (100% dynaaminen):

- 462 looppia, 0 rekursiokutsua

⇒ Lähes 10x parannus, jos ajatellaan että mitä enemmän looppeja, sitä ~~po~~ hitaampi algoritmi (tosiasiassa ei ihan näin, mut tällä on tosi helppo verrata)

Koodin pituus (riviä):

→ ~~Va~~ Vanha: ~165 riviä
→ Uusi: ~ 40 riviä

} poislukien tulostus & vähäinen yhteinen koodi

+++ BOnus:

By adding 6 riviä uutta koodia, sain looppien määrää pienennettyä vielä enemmän.

462 → 246

Tuossa otan siis huomioon sen, että jos esim.

$\begin{matrix} 3 & 3 \\ 4 & 1 \end{matrix}$, niin vierekkäiset numerot kertoo sen, että niissä on tarpeeksi isot yhtenäiset neliöt; ei tarvi mennä ylös ja vasemmalle tarkastaan elementtejä erikseen.

Generalisaatio