CTU Open 2016

Prezentace řešení úloh

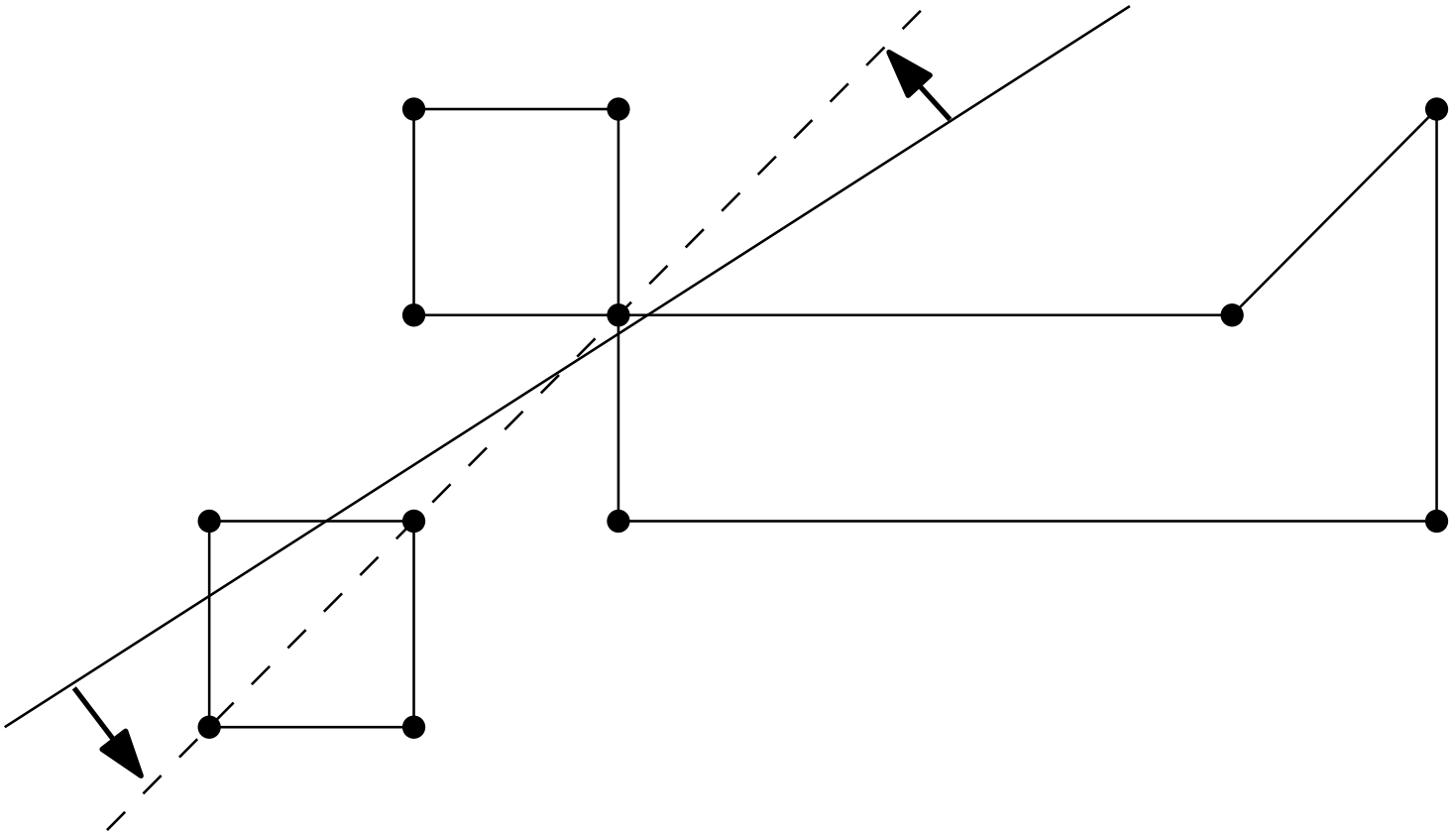
Aerial Archeology

Arch(a)eology

- Vyzkoušet všechny přímky

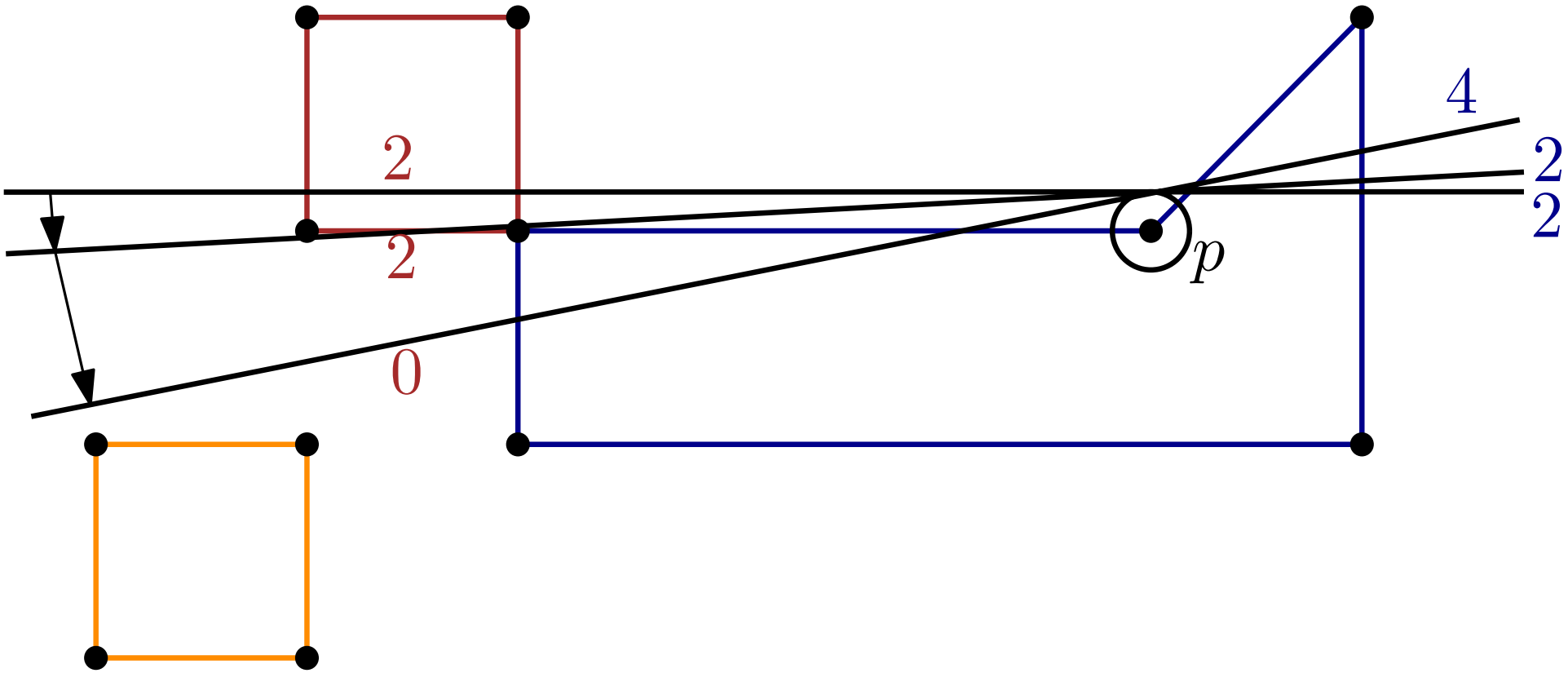
Pomalé řešení

- Stačí přímky těsně vedle dvojice vrcholů
- $O(n^3)$: $O(n^2)$ přímek $O(n)$ práce za každou



Zametací přímka

- Otáčíme okolo každého vrcholu a udržujeme počty průniků s hranicí každého mnohoúhelníka
- O(n^2 log n)



Hot Air Ballooning

Hot Air Ballooning

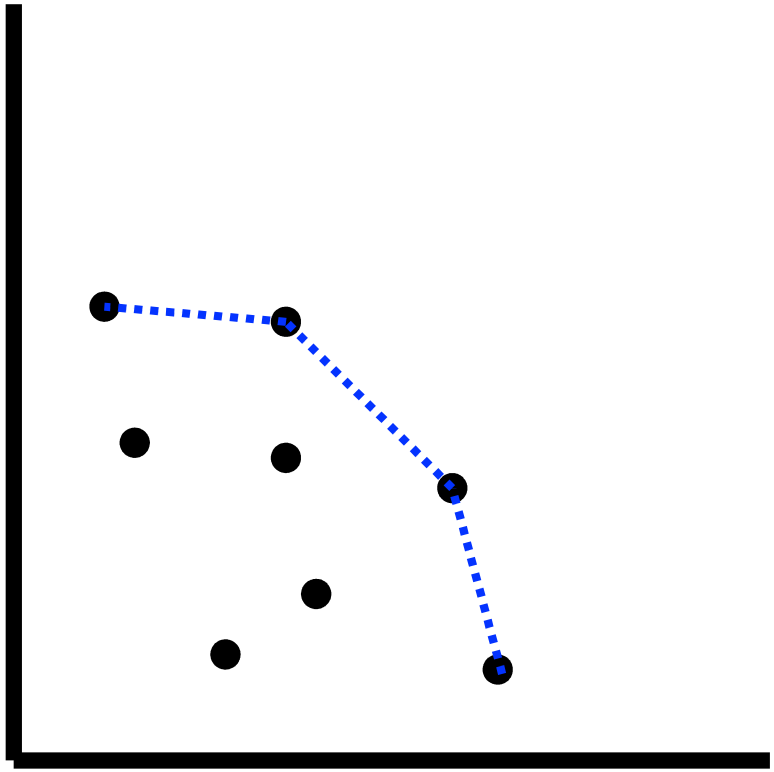
- Načteme seznamy jako čísla.
- Rozdělíme je na číslice a vložíme do množiny.
- Tuto množinu vložíme do množiny množin.
- Vypíšeme velikost množiny množin.
- Lze řešit samozřejmě i jinak.

```
#include <iostream>
#include <set>
using namespace std;

int main(void) {
    int count;
    while(cin >> count) {
       std::set<std::set<int> >
container;
       for(int i=0; i<count; i++) {
          int number;
          cin >> number;
          std::set<int> con_num;
          while(number > 0) {
              int num = number % 10;
              number = number / 10;
              con_num.insert(num);
          }
          container.insert(con_num);
        }
         cout << container.size() << endl;
    }
    return 0;
}
```

Cable Connection

Cable Connection



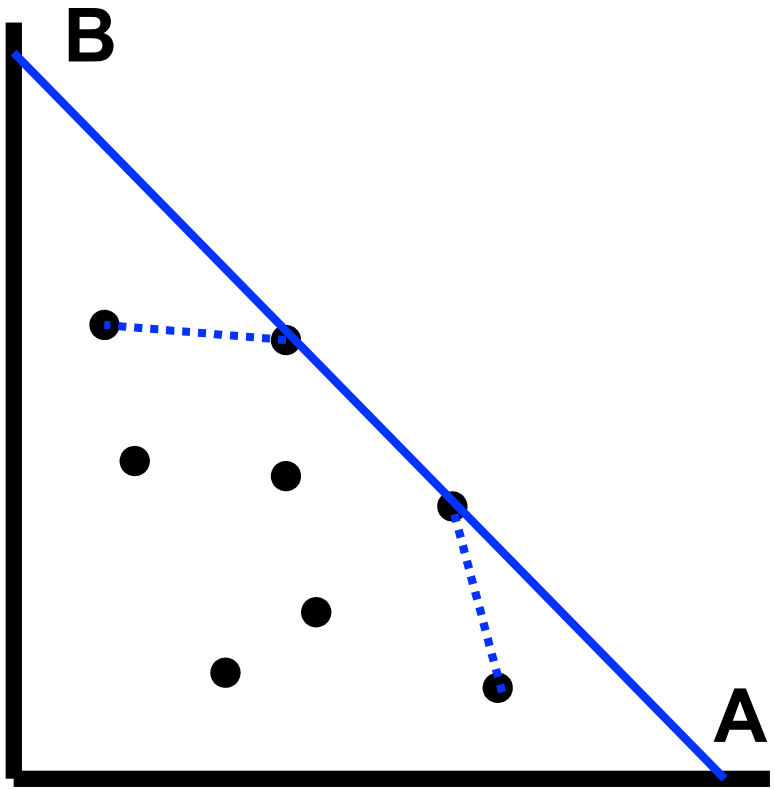
Konvexní obálka,
stačí severovýchodní část
O(N log N)

Možnosti:
A. Kabel jde po hraně obálky
O(N)

B. Kabel prochází jen jedním
bodem obálky
O(N)

Celkem O(N log N)

Cable Connection



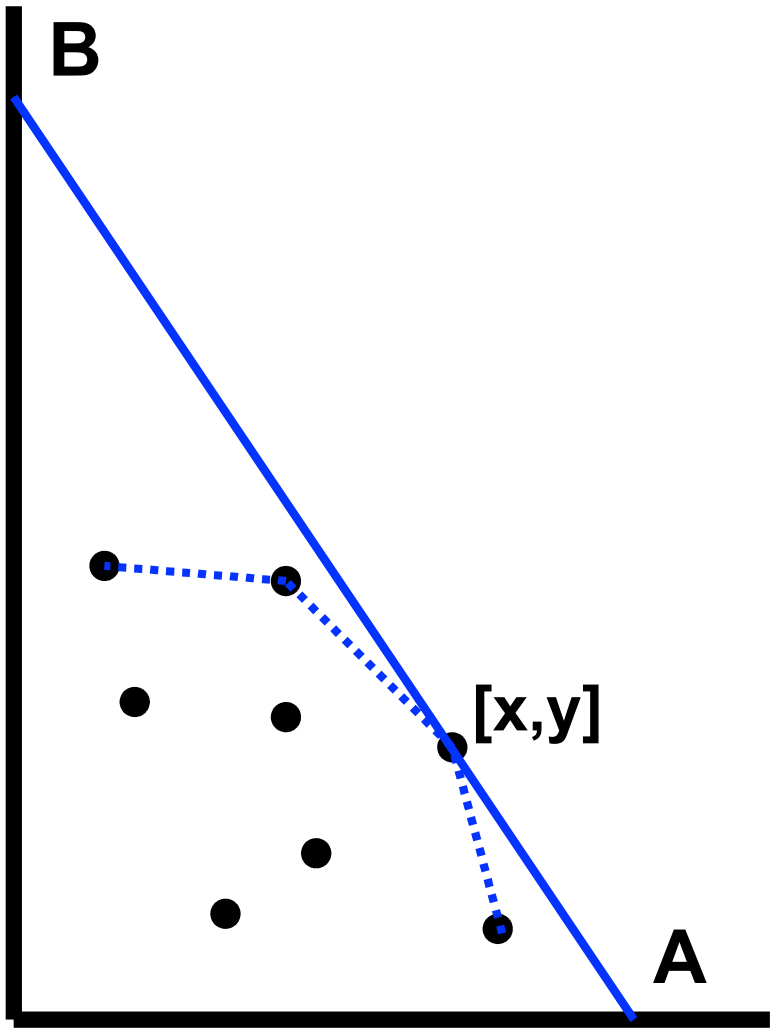
Možnosti:
A. Kabel jde po hraně obálky

Vyzkoušej všechny hrany obálky,
O(N)

Koncové body kabelu A a B pro každou hranu najdeš v čase O(1)

Hranou proložˇ přímku a najdi její průsečíky s osami

Cable Connection



Možnosti:
B. Kabel prochází jen jedním bodem obálky

Vyzkoušej všechny body obálky,
O(N)

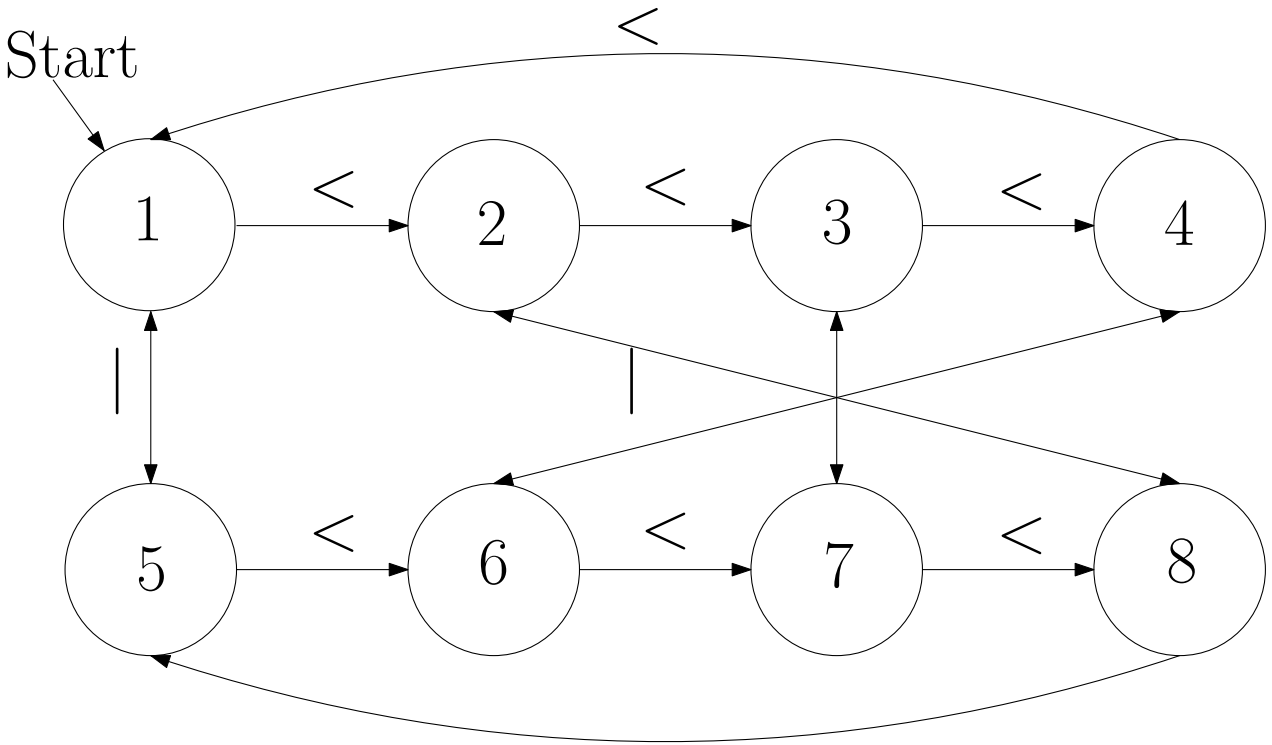
Koncové body kabelu A a B pro každý bod obálky najdeš v čase O(1)

! Analytická geometrie !
! POUŽÍVEJ Taháky !

Display

Display

- Stavový automat s celkem 8 stavy.
- Přechody jsou operace.



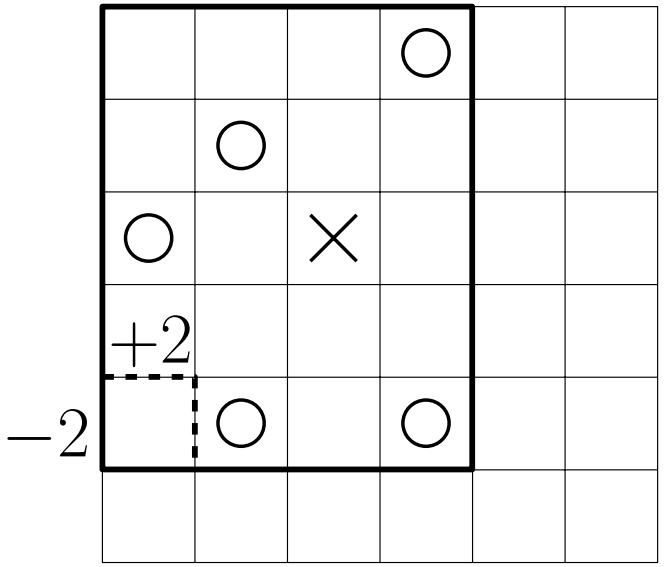
Display

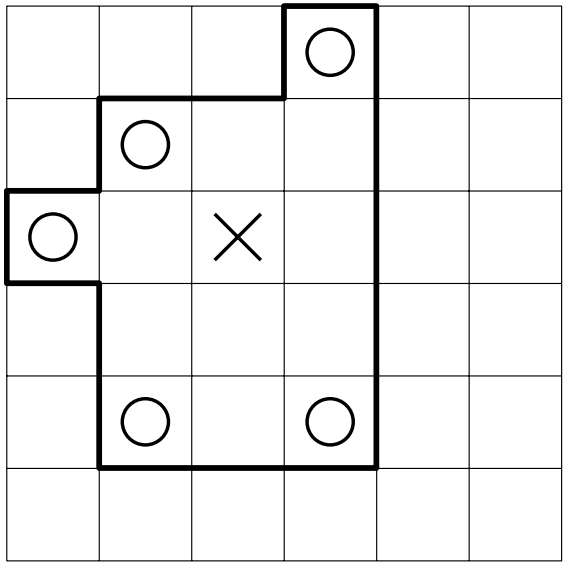
- Stačí implementovat 2 operace např. (<, |) nebo (\, |).
- Ostatní operace lze vyjádřit pomocí dvou implementovaných.

Fence

Ohradíme ovce

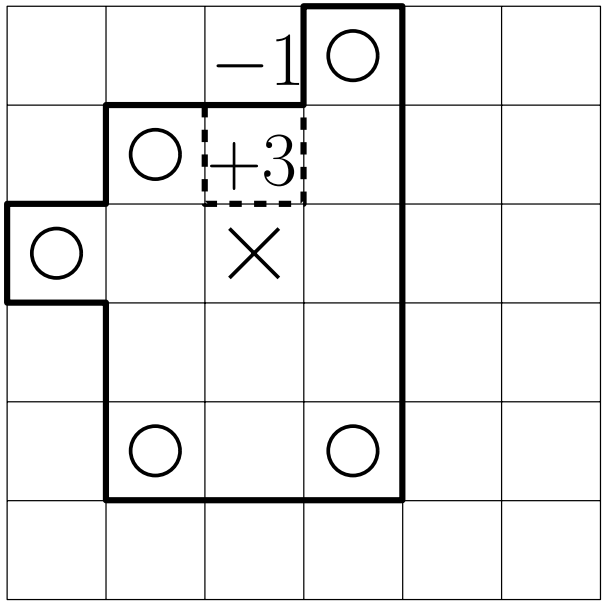
- Nejmenší obvod má například obdélník
- Lokální úpravy
- Stále stejný obvod

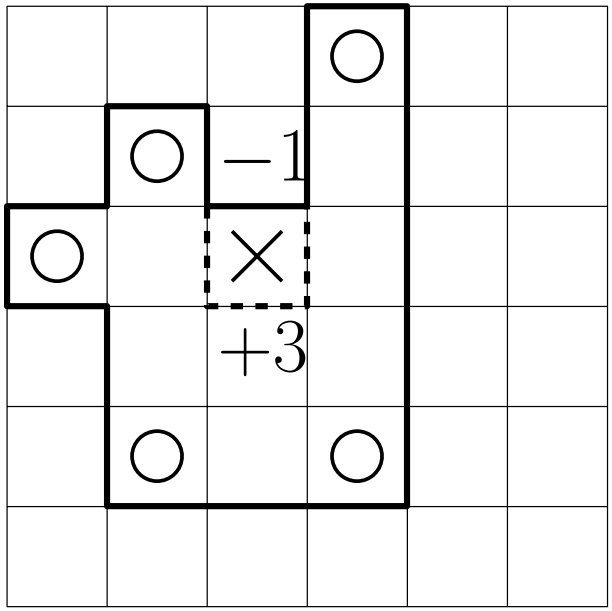




Cesta pro vlka

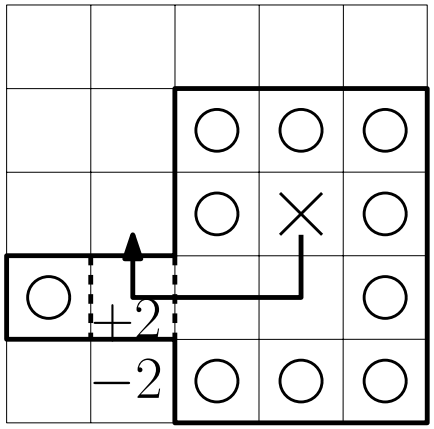
- Každý čtvereček prodlouží ohradu o 2
- Stačí najít nejkratší cestu od vlka ven z ohrady
- Až na...

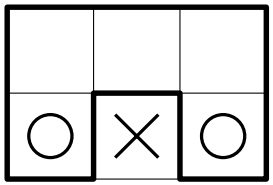


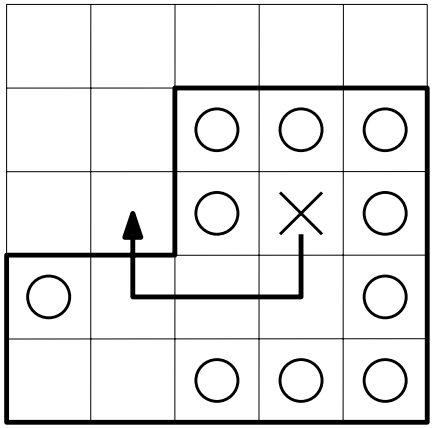


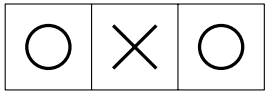
Speciální případy

- Cesta rozdělí ohradu





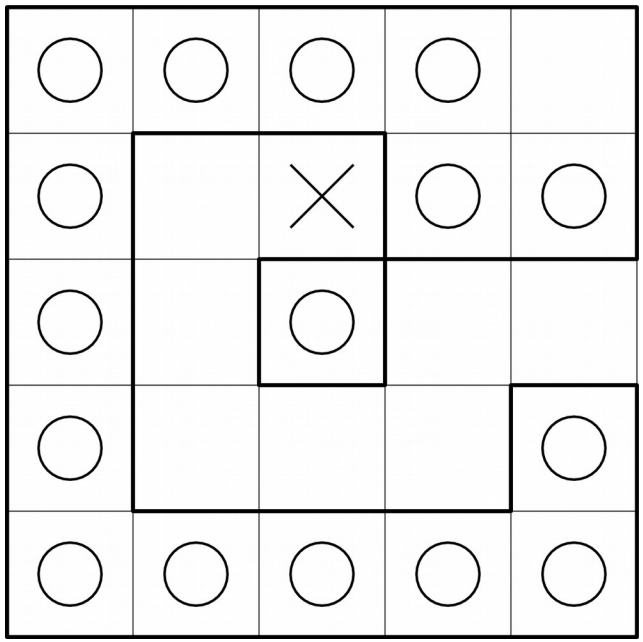




Nemá řešení

Erratum

- Do úlohy se nám vloudila chyba, která byla odhalena až během prezentace řešení.
- Jedná se o to, že v následující situaci vzorová řešení tvrdí, že ohrada existuje, přestože se vlk nemůže dostat ven, pokud se ohrada nesmí sama sebe dotýkat.

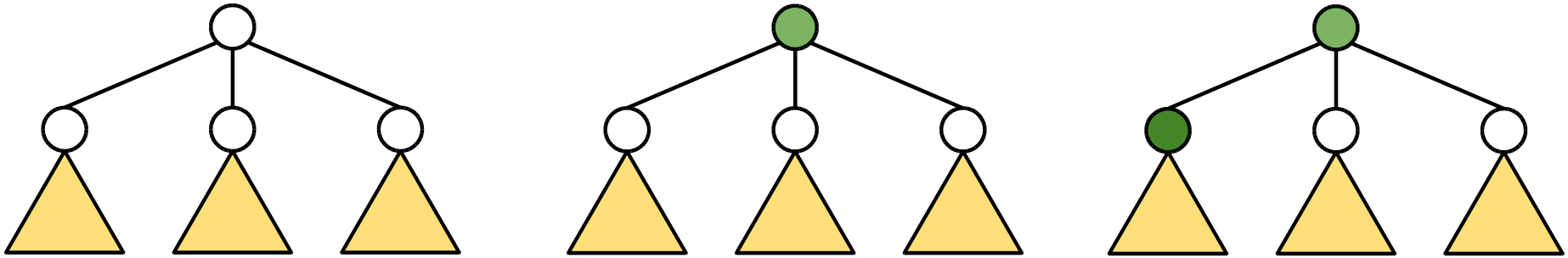


- Za toto nedopatření se omlouváme.
- Další informace naleznete v zadání úlohy.

Tree Stands

Tree Stands

- Requires dynamic programming based solution.
- Keep a 3D table - counts[V][K][S] - keeping the number of ways to place K stands at a subtree rooted at node V, while the state S of the root is either 0 (has no stand), 1 (has safe stand), 2 (has unsafe stand).



- *Tree stand is safe when it is adjacent with yet another tree stand*.

Tree Stands

- Filling up the table entry of counts[V][K][S] while knowing the answers for all descendants of node V has 2 steps.
- First, we compute number of combinations to place K stands in the subtrees of V, while not placing a stand at node V and keeping track of the state of the combinations – aux[K][S]:
  - S = 0 (no descendant has a stand)
  - S = 1 (all stands in descendant nodes are safe)
  - S = 2 (at least one stand at a descendant node is not safe)

Tree Stands

- Using these auxiliary counts, we can get the answer for counts[V][K][S] by:

```
counts[V][0][0] = 1;

for (k = 1; k <= K; k++)
{
    counts[V][k][0] = aux[k][0] + aux[i][1];
    counts[V][k][1] = aux[k-1][0] + aux[k-1][2]);
    counts[V][k][2] = aux[k-1][0];
}
```

- Do not forget to use modulo operator after combining the counts!!!

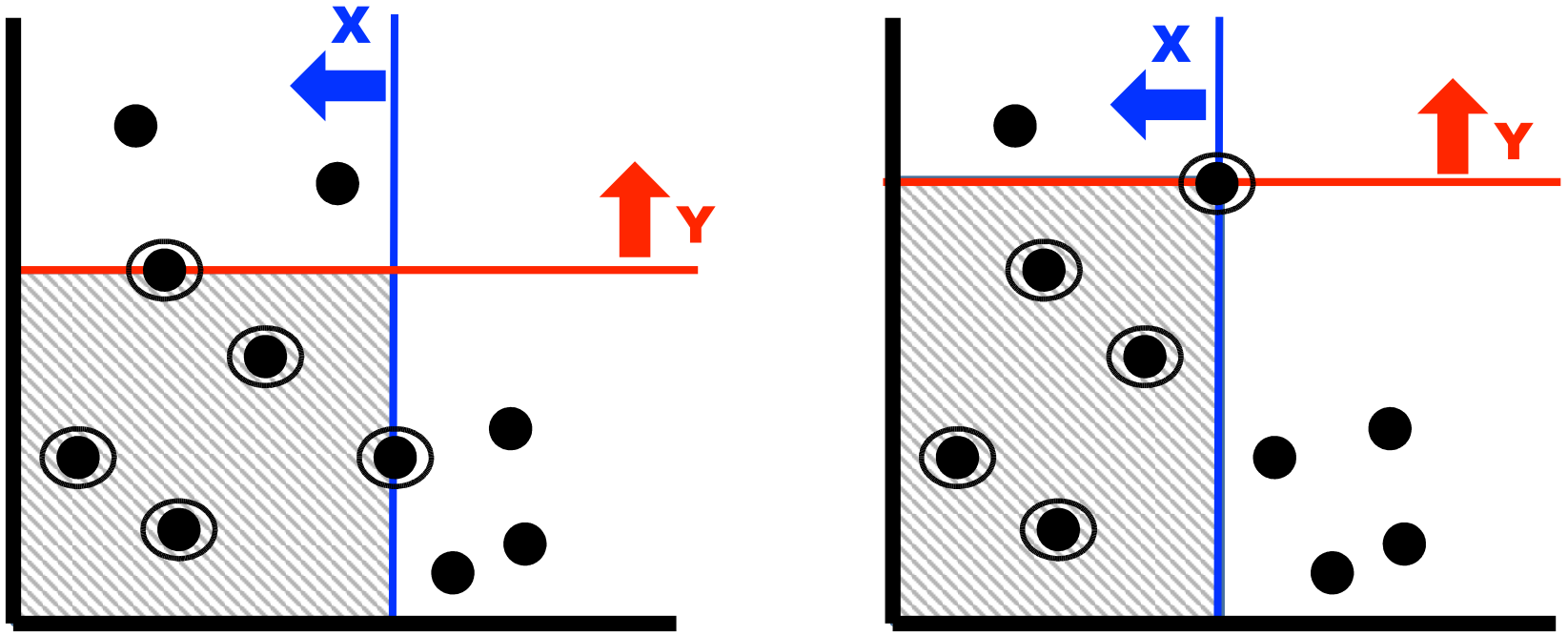
Tree Stands

- Computing the auxiliary counts in aux[K][S] requires yet another DP solution based on already computed counts[U][K][S] for all descendant nodes U of node V.
  - Combining the combinations in the descendant trees need to be done carefully with respect to the states of their trees.
  - See the testers' solutions for more details.

Orchard

Orchard Division

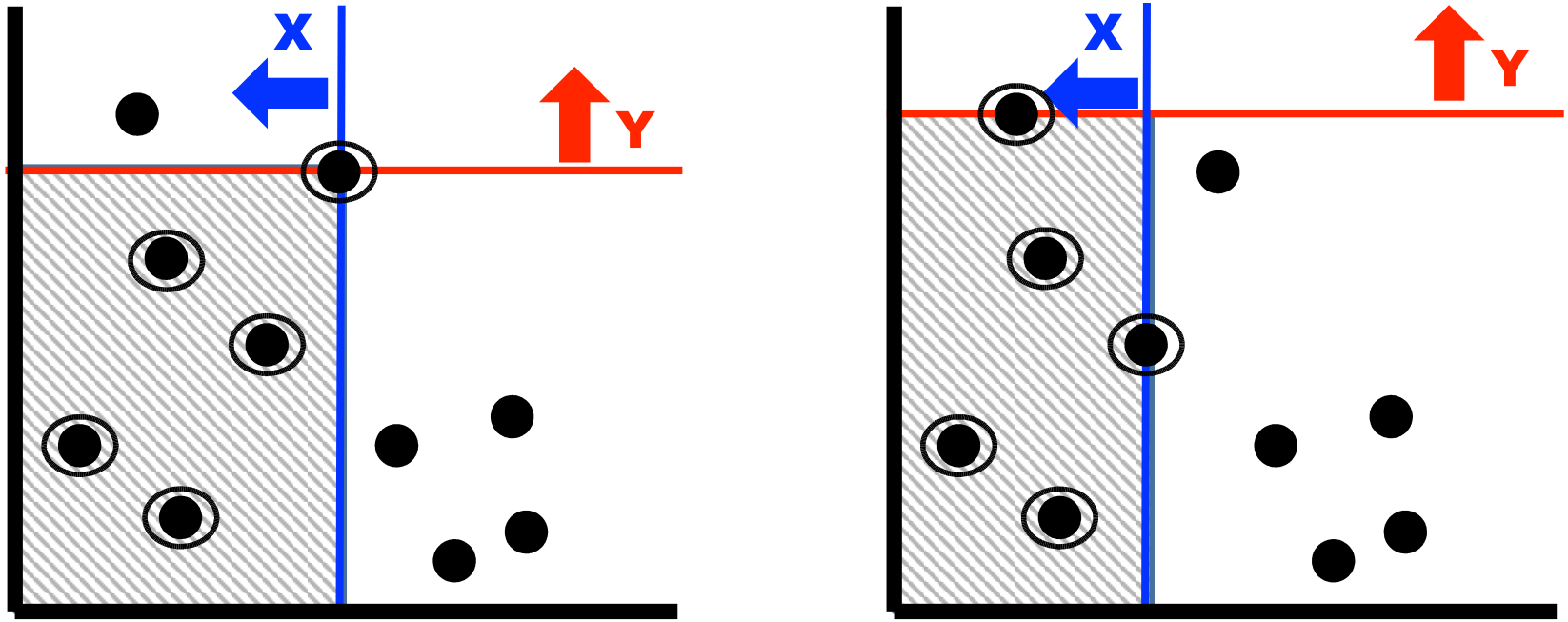
Dvě zametací přímky X a Y na sebe kolmé určují obdélník s kandidáty na opt. řešení



V jednom kroku
posuň Y nahoru k další obsazené y souřadnici
snaž se maximálně zmenšit X a udržet kandidáty

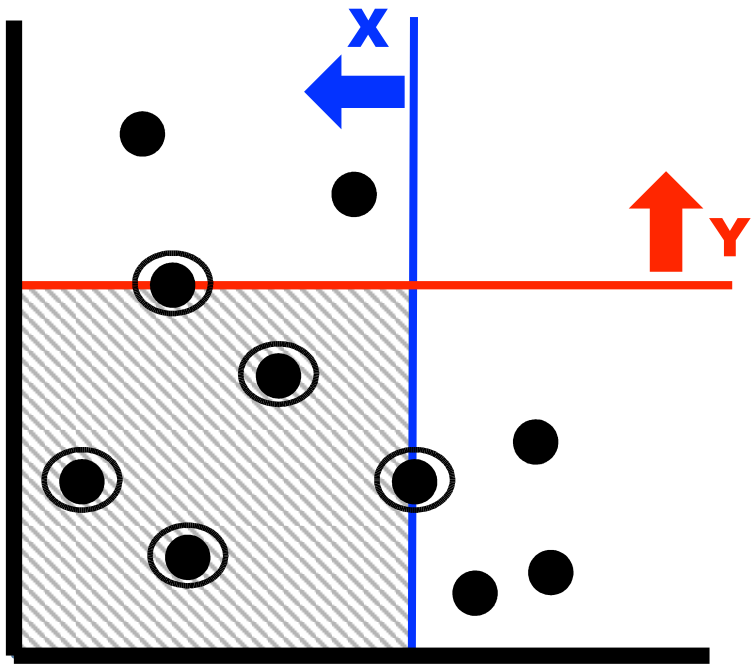
Orchard Division

Dvě zametací přímky X a Y na sebe kolmé určují obdélník s kandidáty na opt. řešení



V jednom kroku
posuň Y nahoru k další obsazené y souřadnici
snaž se maximálně zmenšit X a udržet kandidáta

Orchard Division

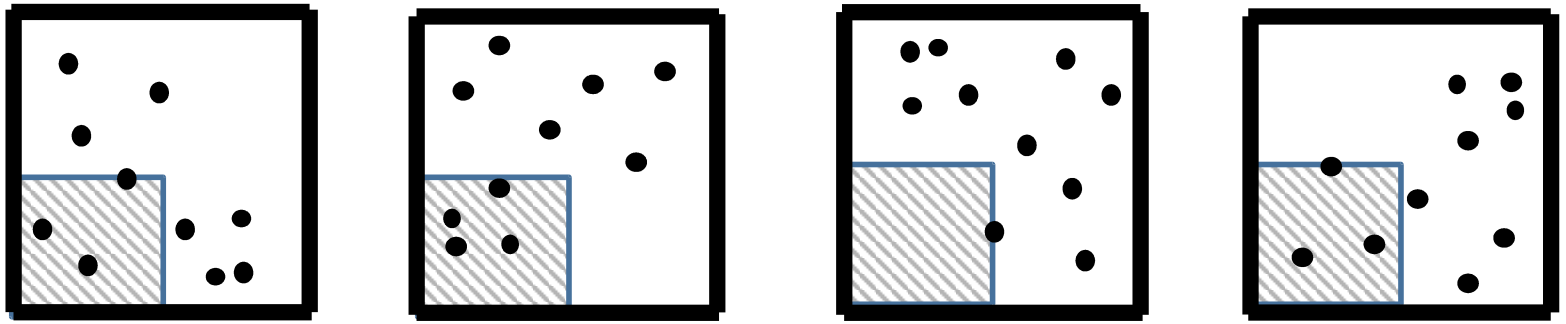


Kandidátní body ukládej do max-prioritní fronty podle x souřadnice

Zohledni možnou vertikální a horizontální kolinearitu

Amortizovaná složitost jednoho kroku O(log N).

Orchard Division



Čtyři možné rohy, čtyři rotace matice, stejný kód řešení.

Celkem čtyřikrát,
pokaždé i s úvodním seřazením podle y souřadnice

O(N log N)

Orchard Division

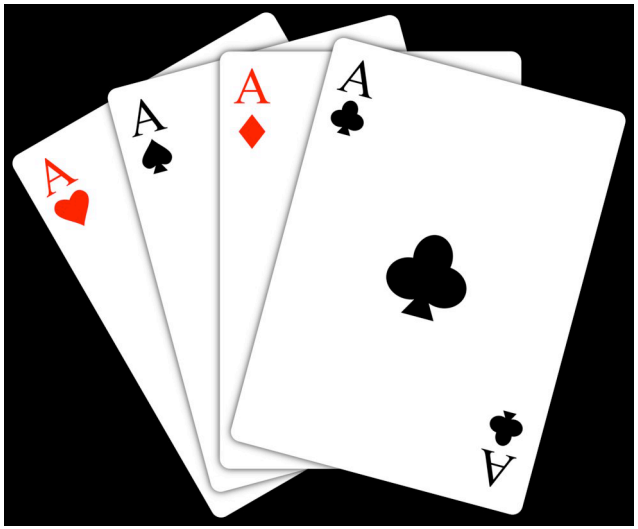
Není to tak těžké, celý kód solveru:

```
long long solve() {
    sort(ps, ps+N);
    priority_queue<int> q;
    long long result = -1;
    int x = M;

    for (int i = 0; i < N; ) {
        int y = ps[i].y;
        for (; i < N && ps[i].y == y; ++i)
            if (ps[i].x < x)
                q.push(ps[i].x);
        while (q.size() * 2 > N) {
            x = q.top();
            while (!q.empty() && q.top() == x)
                q.pop();
        }
        if (q.size() * 2 == N) {
            long long area = (long long)(y + 1) * (q.top() + 1);
            if (result == -1 || area < result) {
                result = area;
            }
        }
    }
    return result;
}
```

It's Raining, Man

Raining



- Karta = vrchol,
- Stejná barva nebo hodnota = hrana

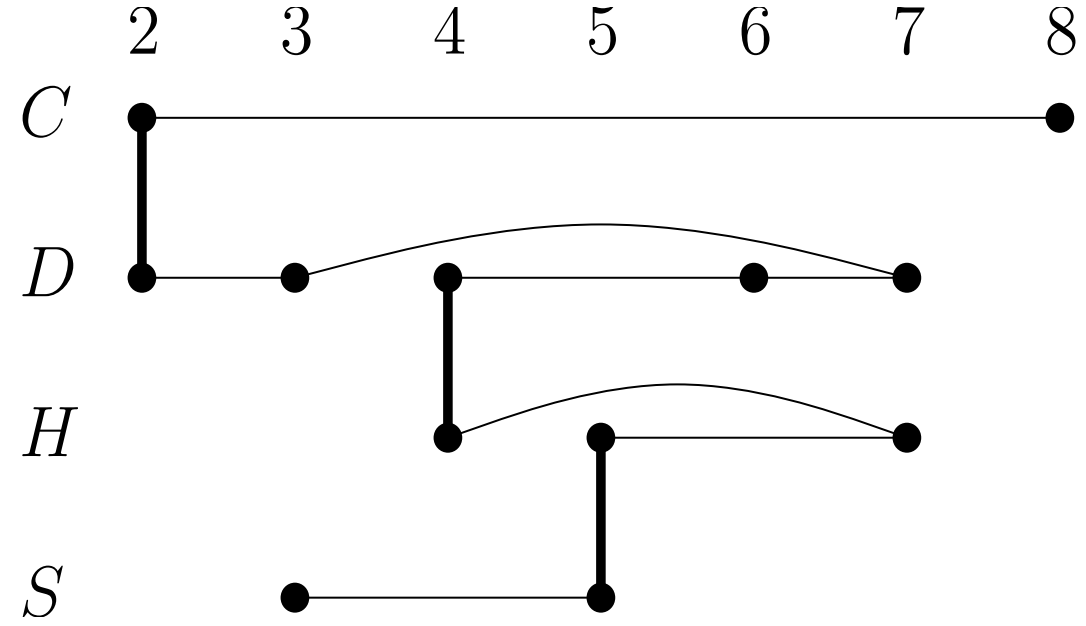
→ Hamiltonovská cesta na až 52 vrcholech (příliš)

- Graf je ale speciální

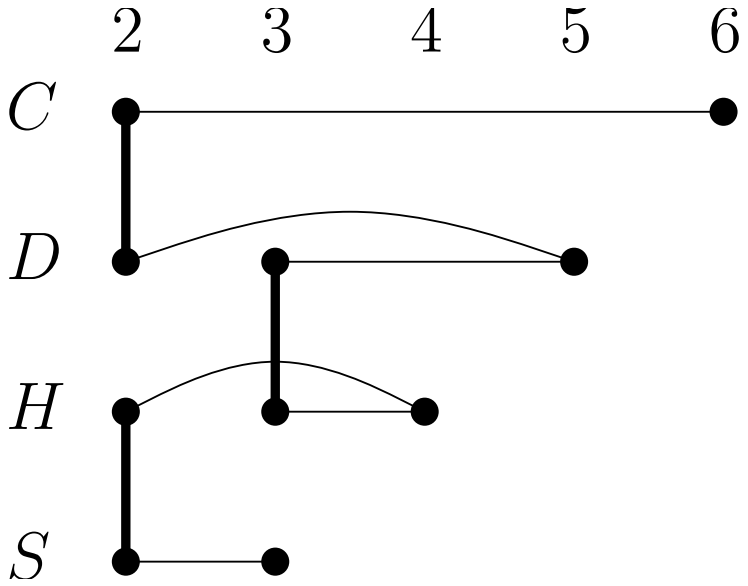
→ Rozbor případů

Projdeme jednu barvu po druhé

- Stačí najít cestu mezi barvami

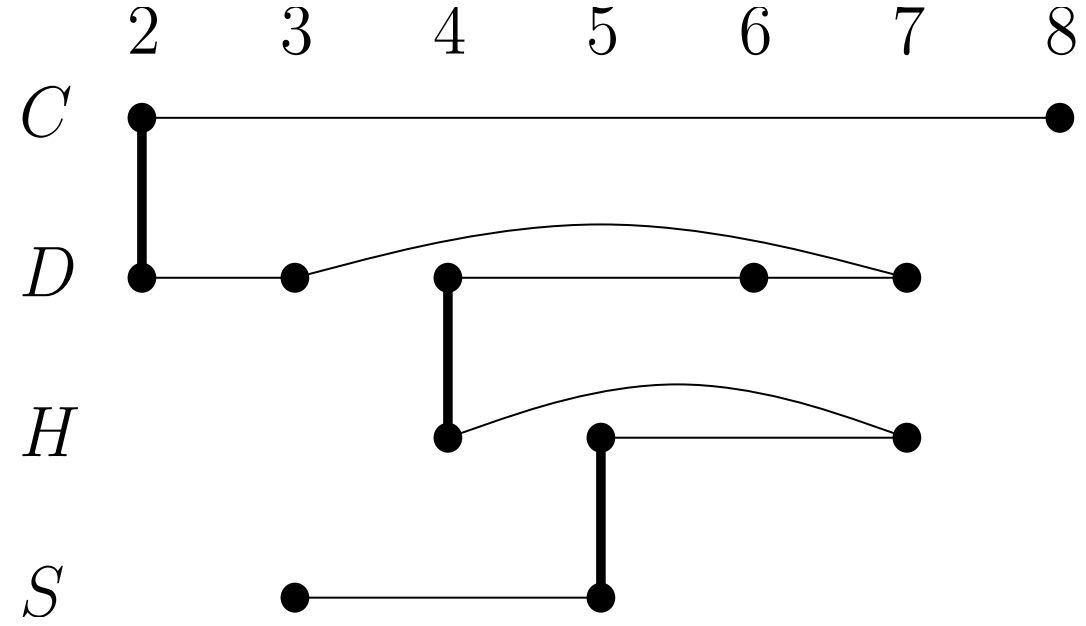


- I toto je cesta

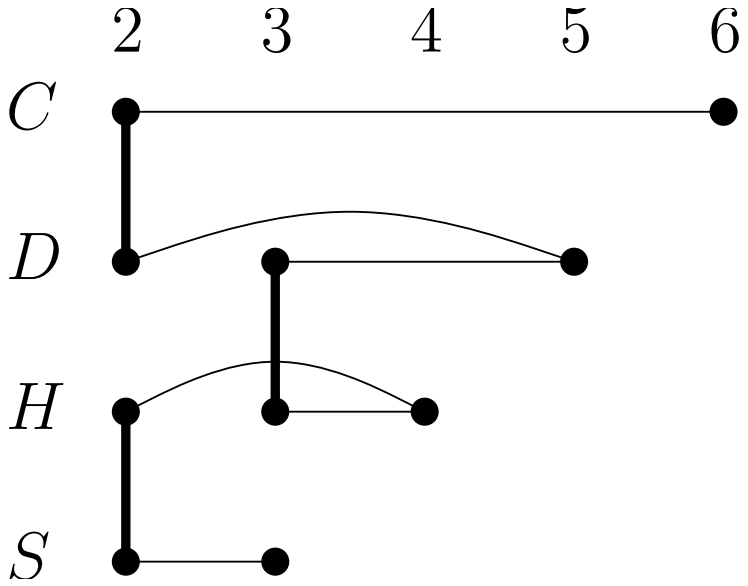


Projdeme jednu barvu po druhé

- Stačí najít cestu mezi barvami

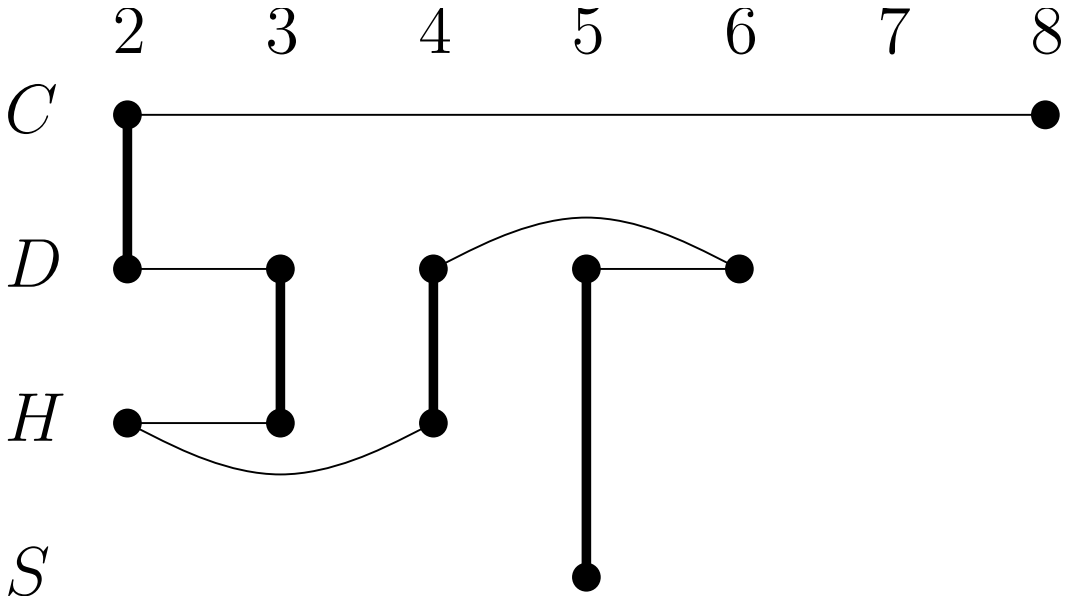


- I toto je cesta

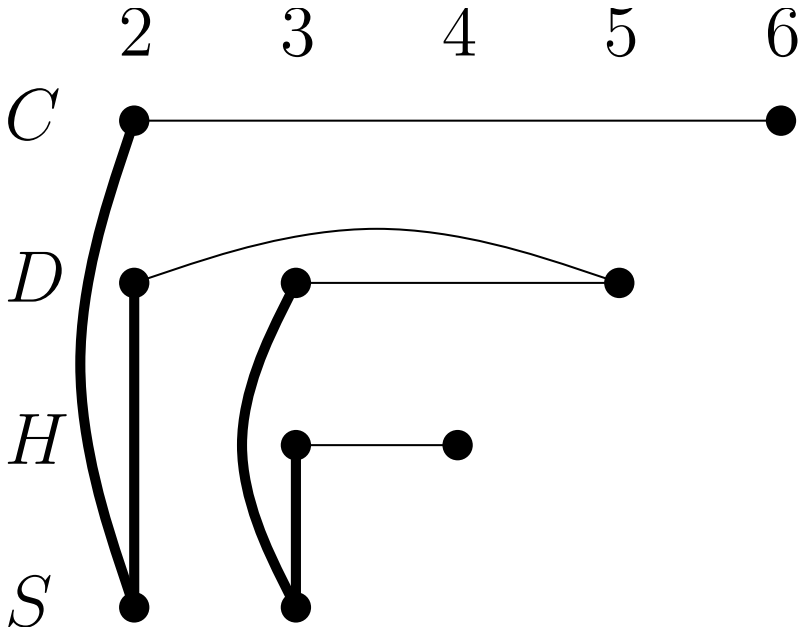


Další možnosti

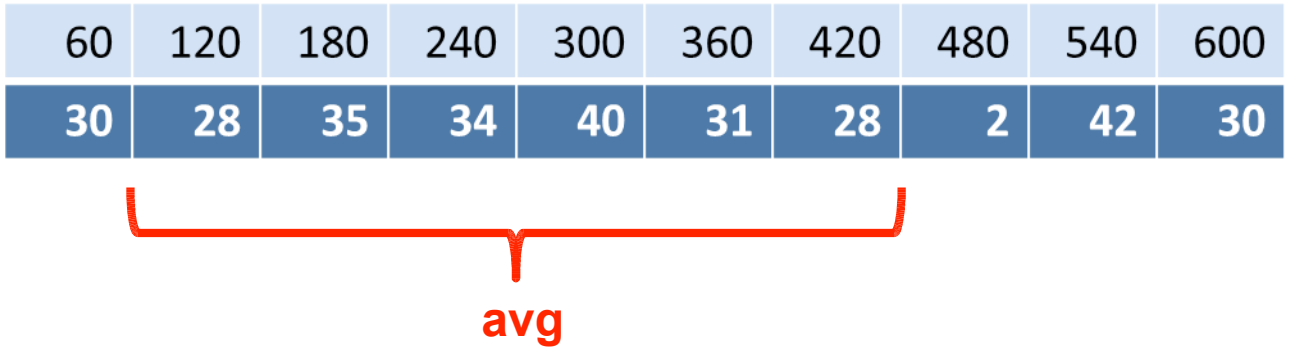
- Centralizované (v D)



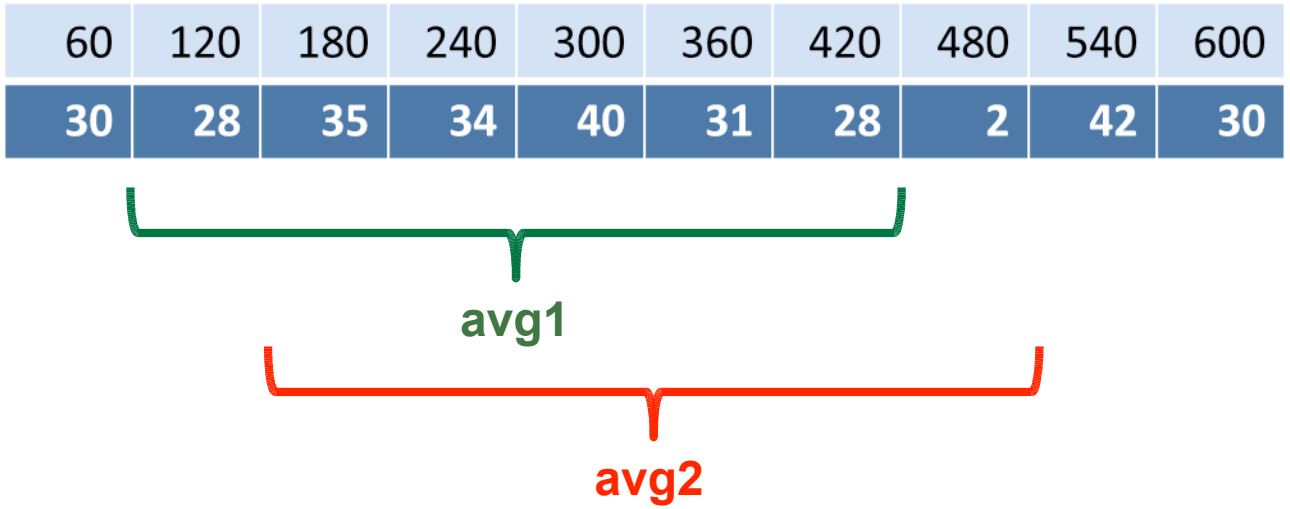
- Úskoky (do S)



Suspicious Samples

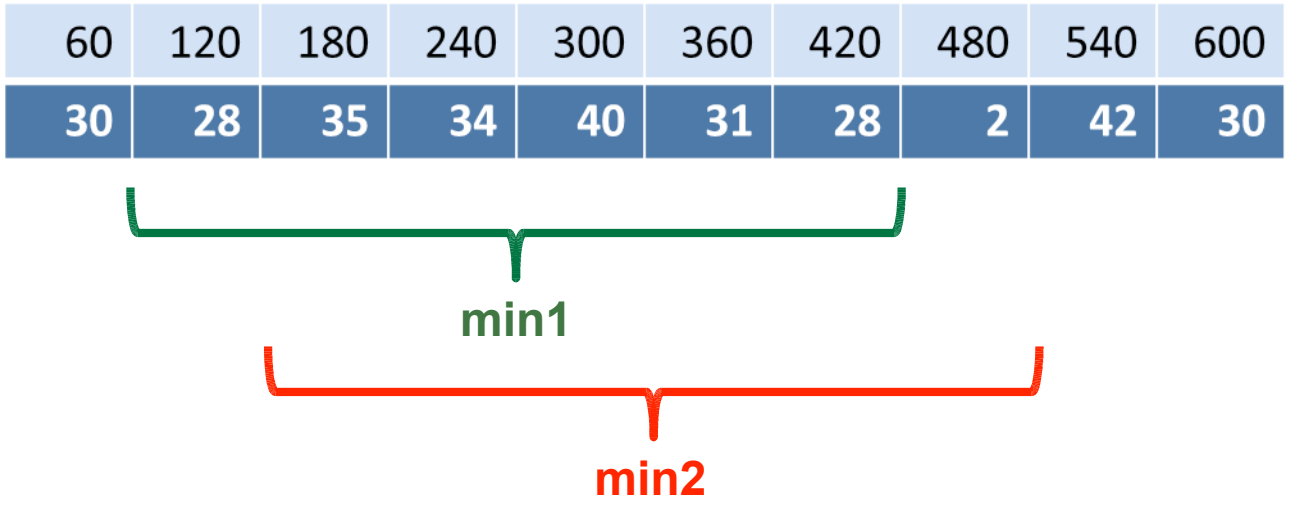



```
int sum = 0;
for (int i=left; i<right; ++)
  sum += value[i];
return sum / (right-left);
```




```
sum2 = sum1
     + value[right++]
     - value[left++];
```

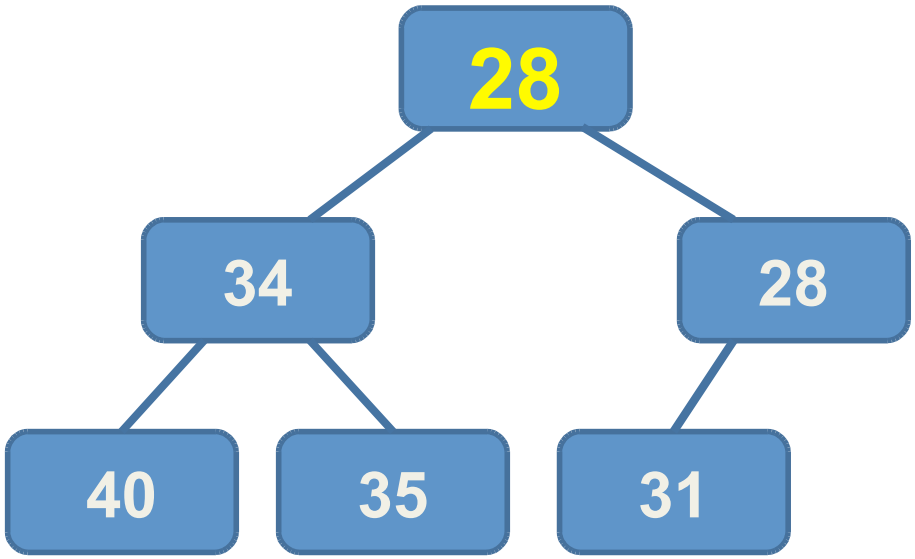
Klouzavý průměr (moving average)


60
120
180
240
300
360
420
480
540
600
30
28
35
34
40
31
28
2
42
30
min1
min2

60	120	180	240	300	360	420	480	540	600
30	28	35	34	40	31	28	2	42	30

min1

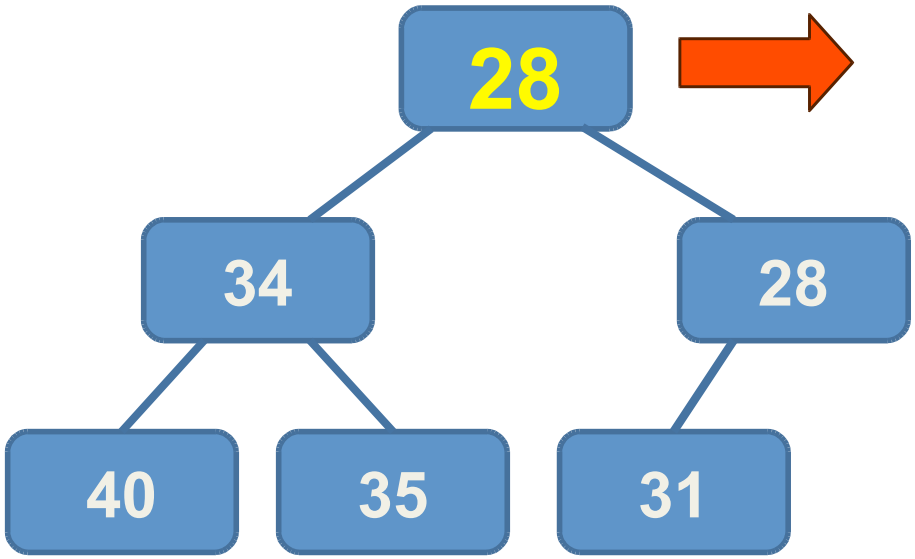
min2

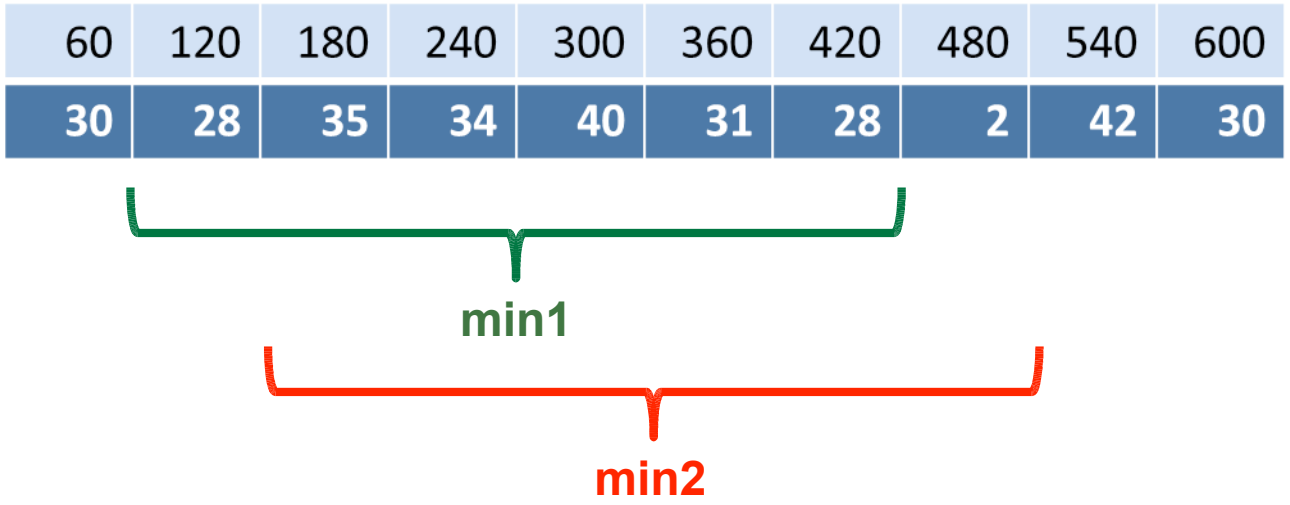


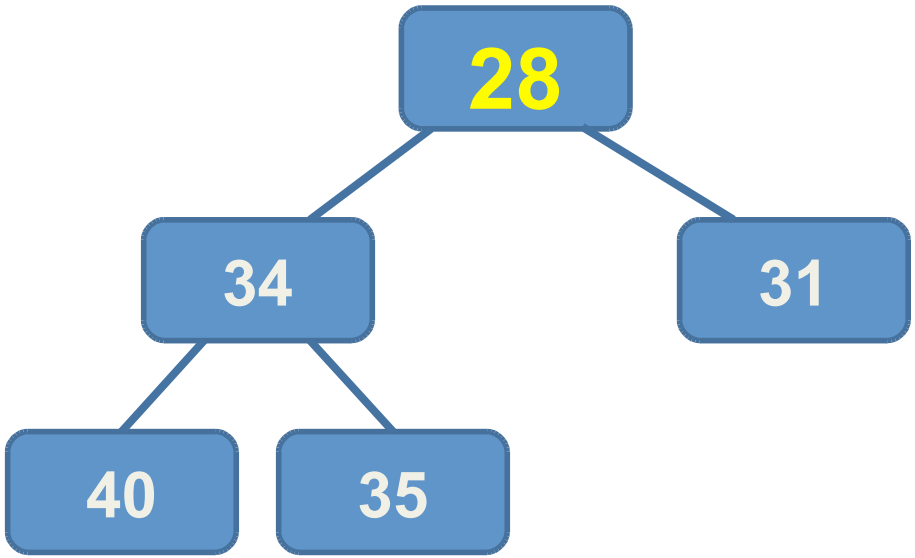
60	120	180	240	300	360	420	480	540	600
30	28	35	34	40	31	28	2	42	30

min1

min2




60 120 180 240 300 360 420 480 540 600
30 28 35 34 40 31 28 2 42 30
min1
min2


28
34
31
40
35

Colorful Tribune

Tribune

Latinský čtverec řádu N

1	2	3	4	5
5	1	2	3	4
4	5	1	2	3
3	2	5	1	2
2	3	4	5	1

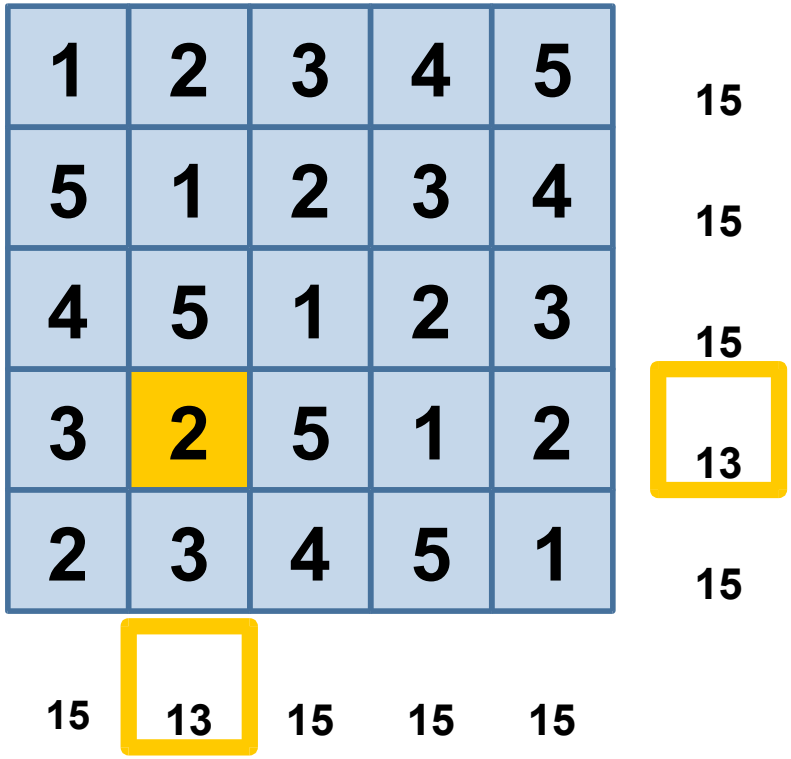
Pravidlo: Každý řádek i sloupec obsahuje všechny symboly z daných N symbolů a každý z nich právě jednou

Hledaný prvek porušuje toto pravidlo

Vylož si každý symbol (v zadání: barvu) jako celé číslo

Tribune

Latinský čtverec řádu N



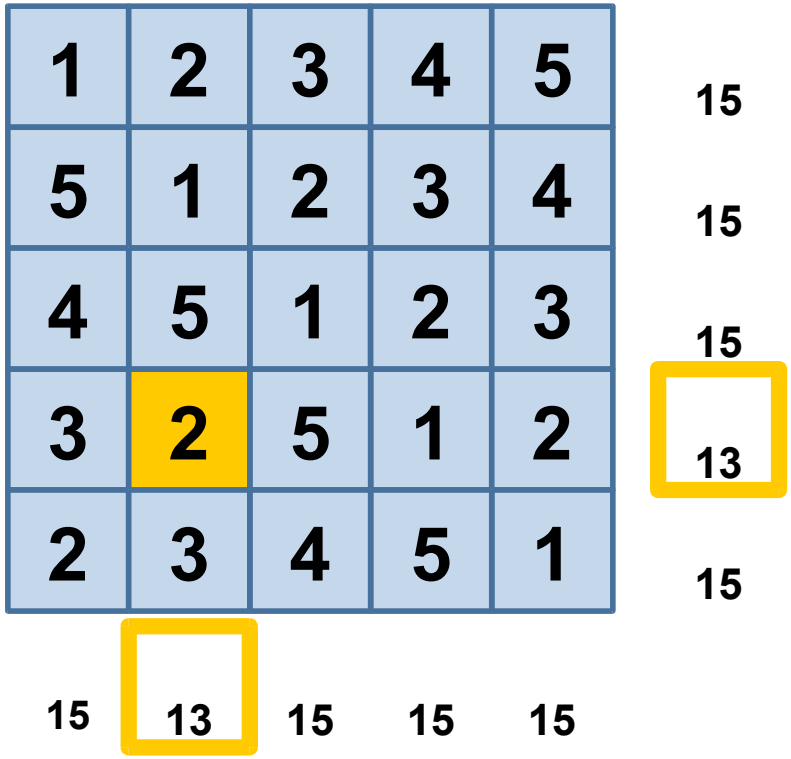
Odlišné řádkové a sloupcové součty určují řádek a sloupec hledaného prvku

Správná hodnota hledaného prvku je
Správný součet *minus* odlišný součet *plus* hodnota odlišného prvku
Např. 4 = 15 — 13 + 2

Nemusíme testovat hodnoty žádných prvků
Stihneme za jeden průchod
$O(N^2)$

Tribune

Latinský čtverec řádu N



Odlišné řádkové a sloupcové součty určují řádek a sloupec hledaného prvku

Správná hodnota hledaného prvku je
Správný součet *minus* odlišný součet *plus* hodnota odlišného prvku
Např. 4 = 15 — 13 + 2

Nemusíme testovat hodnoty žádných prvků
Stihneme za jeden průchod
$O(N^2)$

Problem	OK	WRONG	FIRST
Archeology	0	1	
Baloon	83	128	0:06
Cable	3	7	2:49
Display	53	24	1:41
Fence	0	5	
Huntsmen	5	10	1:51
Orchard	4	45	3:03
Raining	0	80	
Samples	18	164	1:23
Tribune	75	150	0:23
	241	614	