Korešpondenčný seminár z programovania

Leták letnej časti XLI. ročníka

Korešpondenčný seminár z programovania (KSP) je súťaž programátorov – stredoškolákov a mladších – pripravovaná skupinou študentov Fakulty matematiky, fyziky a informatiky Univerzity Komenského v Bratislave. Naším cieľom je zdokonaliť žiakov v programovaní a v algoritmickom myslení.

Riešením súťažných úloh a štúdiom vzorových riešení sa zlepšíš v programovaní a naučíš sa algoritmicky rozmýšľať. Získané poznatky a skúsenosti využiješ v iných súťažiach v programovaní (napríklad pri riešení Olympiády v informatike), v bežnom živote, počas vysokoškolského štúdia, dokonca aj na prijímacích pohovoroch do zamestnania. Naši riešitelia sa každoročne zúčastňujú a úspešne umiestňujú na medzinárodných olympiádach v informatike (v Austrálii, Taliansku, Kazachstane, Taiwane, …). Mnoho našich bývalých riešiteľov sa tiež bez ťažkostí zamestnalo v špičkových IT spoločnostiach ako Google, Facebook, ESET, …

Ak študuješ na strednej škole a zaujíma ťa programovanie, neváhaj a zapoj sa do KSP:

Ako sa zapojiť do KSP?

- **Prečítaj** si zadania. Nájdeš ich v tomto letáku a na našej stránke https://www.ksp.sk/ulohy. Každý rok máme zimnú a letnú časť, obe majú dve kolá s ôsmimi úlohami.
- Ťeš sa, aké sú tento rok pekné úlohy.
- **Vyrieš** úlohy. Nemusíš vyriešiť všetky, nemusíš ich vyriešiť najlepšie ako sa dá. Aj za čiastočné riešenia sa dostávajú body, za každú úlohu za dá získať 0 až 20 bodov.
- Na riešenie úloh jedného kola máš približne dva mesiace a môžeš ich riešiť doma bez toho, aby si niekam cestoval. Termín odovzdania úloh je napísaný aj na našej stránke, aj v PDF zadaniach. Úlohy sa nedajú odovzdávať po termíne, takže si to, prosím, nenechaj na poslednú chvíľu.
- Úlohy rieš samostatne a neprezrádzaj riešenia ostatným riešiteľom. Odpisovanie riešení a prezradenie riešení pred termínom kola je porušením pravidiel KSP. Po skončení kola sa, samozrejme, o riešeniach rozprávať môžeš. :)
- **Odovdzaj** riešenia úloh. Odkaz na odovzdávanie úloh nájdeš pod webovým zadaním každej úlohy alebo na stránke https://www.ksp.sk/odovzdavanie. Na odovzdávanie sa treba prihlásiť, aby sme vedeli, komu máme dať body.
  - Vo väčšine úloh odovzdávaš program a popis.
  - Program je hneď po odovzdaní otestovaný testovačom a hneď vidíš, koľko bodov za program máš. Program môžeš odovzdávať znova a znova, až kým nie si spokojný s výsledkom. Ak nevieš, ako majú vyzerať odovzdané programy, pozri si https://www.ksp.sk/odovzdavanie-programov
  - Do popisu slovne napíšeš, ako tvoje riešenie funguje, prečo funguje a tiež odhad časovej a pamäťovej zložitosti programu. Viac sa dozvieš na stránke https://www.ksp.sk/ako-riesit. Popis opravia a obodujú vedúci KSP po skončení kola.
- Po skončení kola si **prečítaj vzorové riešenia** úloh (veľa sa z toho naučíš), pozri svoje opravené popisy (či ti tam vedúci nenapísali nejaké poučné komentáre), pozri sa do výsledkovky a **teš sa**, koľko máš bodov. Vo výsledkoch sa hodnotí samostatne letná a zimná časť. V každej časti je dôležitý celkový súčet bodov.
- Prečo sa máš tešiť z bodov? Čítaj ďalej.

Čo môžem vyhrať?

- Okrem neoceniteľných vedomostí, skúseností a zručností, ktoré získaš pri riešení semináru, môžeš vyhrať množstvo skvelých vecí.
- Všetci víťazi od nás dostanú **vecné ceny**.
- Pre 36 najlepších riešiteľov organizujeme každoročne dve týždenné **sústredenia**. Sústredenie je niečo ako tábor, na ktorom spoznáš nových priateľov s podobnými záujmami, naučíš sa čosi viac nielen o programovaní a zažiješ kopec zábavy. Sústredenia sú fakt skvelé akcie, najmä, keď ich organizuje Trojsten.

- Aby ste sa mohli pochváliť ostatným, akí ste šikovní, víťazom všetkých levelov udelíme a pošleme **diplomy**.
- Aj keď sa nedostaneš medzi víťazov, stále môžeš byť úspešným riešiteľom. Úspešný riešiteľ je ten, kto získal aspoň polovicu bodov počas celej časti (letnej, či zimnej).

Pravidlá a levely

Počnúc tridsiatym piatym ročníkom rušíme staré kategórie a prechádzame na nový systém *levelov*.

Každý riešiteľ má level, číslo od 1 po 4. Noví riešitelia začínajú na leveli 1 a pokiaľ sa im v riešení darí, level im postupne rastie. Svoj level si môže každý riešiteľ pozrieť na našej stránke. Riešiteľom s levelom L sa započítavajú body len za úlohy s číslami L až 8.

Vo výsledkových listinách (https://www.ksp.sk/vysledky) sa každému riešiteľovi počíta **5 najlepšie vyriešených úloh**. Celkovo sa dá za časť (dve kolá) získať 200 bodov. Riešitelia, ktorí sa v nejakej výsledkovke umiestnili na jednom z prvých dvoch miest a majú aspoň 150 bodov sú **víťazi**. Najlepších 36 riešiteľov pozývame na sústredenie.

Podrobnejšie pravidlá si môžete prečítať na https://www.ksp.sk/pravidla.

Registrácia

Pred odovzdaním riešenia je potrebné sa zaregistrovať na našej webstránke a vyplniť požadované kontaktné údaje. Odporúčame sa zaregistrovať aspoň pár dní pred odovzdávaním riešenia (pre prípad, že by ste mali počas registrácie nejaké problémy).

Účasťou v KSP nám dávate súhlas spracovať a archivovať údaje, ktoré nám poskytnete pri registrácii, ako aj zverejniť vaše meno, školu, ročník a získané body vo výsledkovej listine.

Korešpondenčný seminár z programovania
XLI. ročník, 2023/24
FMFI UK,
Mlynská Dolina, 842 48 Bratislava

Úlohy 2. kola letnej časti

Termín odoslania riešení tohto kola je pondelok **17. júna 2024**. Doprogramovávanie končí v pondelok 1. júla 2024.

1. Divný sen

12 b za popis, 8 b za program

Ferkovi sa sníval hrozne zvláštny sen. Vo sne šoféroval po hrboľatej diaľnici. Za oknami sa mihala ubiehajúca krajina a Ferko uháňal na plný plyn. Potom zrazu niekde zastal, vytiahol lopatu a začal rovno pri ceste kopať. Za chvíľu vykopal truhlicu a v nej bol obrovský zlatý poklad.

Keď sa Ferko zobudil, bolo mu jasné že ten sen musí byť prorocká vízia. Vytiahol atlas, našiel v ňom najbližšiu diaľnicu a pustil sa hľadať, kde by ten poklad mohol byť. Ale ako to pri snoch bohužiaľ býva, všetky detaily okamžite zabudol a zostali mu v pamäti len hmlisté obrysy.

Pomôžte Ferkovi spomenúť si, kde má kopať!

Úloha

Mapa hrboľatej diaľnice je rozdelená na n kilometrov. O každom kilometri mapa hovorí, aký je strmý, čiže o koľko metrov stúpne alebo klesne naša nadmorská výška ak ním prejdeme. Tiež o každom kilometri vieme, či tam diaľnica vedie cez les alebo cez pole.

Ferko vo sne prešiel nejaký súvislý úsek diaľnice. Ale vôbec si nevie spomenúť, koľko kilometrov dokopy prešiel, na akom mieste začal, ani kde skončil. Pamätá si iba zopár vecí:

- Išiel smerom preč od svojho mesta (nie v protismere).
- Začiatok aj koniec boli pri nejakom diaľničnom stĺpiku, čiže na celočíselnej pozícii.
- Na začiatku sna išiel presne jeden kilometer cez les. Odvtedy už nikdy znova nešiel cez les.
- Na konci cesty bola Ferkova nadmorská výška presne o s metrov vyššie, ako na začiatku. (Ak je s záporné, skončil nižšie.)

To ešte stále nemusí byť jednoznačné. Preto spočítajte počet možností, kde sa mohol sen odohrať.

Dve možnosti považujeme za rôzne, ak začínajú alebo končia na inom mieste.

Formát vstupu

Na prvom riadku je celé číslo s udávajúce, o koľko dokopy stúpla/klesla Ferkova nadmorská výška počas jeho vysnívanej cesty.

Na druhom riadku je kladné celé číslo n udávajúce celkovú dĺžku diaľnice v kilometroch.

Ďalej nasleduje n riadkov obsahujúcich dve celé čísla v_i a k_i. v_i určuje zmenu nadmorskej výšky na i-tom kilometri. k_i sa rovná buď 1 ak je na i-tom kilometri les, alebo 0 ak je tam pole.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2
$1 \leq n \leq$	100	100 000
$s \geq$	$-10\,000$	-10^{11}
$s \leq$	10 000	10^{11}
$v_i \geq$	-100	-10^6
$v_i \leq$	100	10^6

Ak programujete v jazyku C++, dajte si pozor na veľkosť číselných premenných. Použite radšej `long long` ako `int`.

Formát výstupu

Vypíšte jedno nezáporné celé číslo: počet rôznych úsekov diaľnice, ktoré sa mohli Ferkovi snívať.

Príklady

vstup

```
9
10
1 0
8 0
4 1
5 0
7 1
1 0
20 1
-11 0
0 0
2 0
```

výstup

```
3
```

Vo sne mohol prejsť tretí a štvrtý kilometer ($4 + 5 = 9$), alebo siedmy a ôsmy kilometer ($20 - 11 = 9$), alebo siedmy až deviaty kilometer ($20 - 11 + 0 = 9$).

vstup

```
5
8
2 1
3 0
4 0
0 0
-4 0
0 0
3 1
5 1
```

výstup

```
4
```

Možnosti sú $2 + 3 = 5$, $2 + 3 + 4 + 0 - 4 = 5$, $2 + 3 + 4 + 0 - 4 + 0 = 5$ alebo $5 = 5$.

vstup

```
0
3
0 0
0 0
0 0
```

výstup

```
0
```

Diaľnica vedie po rovine, čo by síce sedelo, ale ide iba cez polia – les nikde. Predsalen to nebol prorocký sen.

2. Incident na diaľnici

12 b za popis, 8 b za program

Na diaľnici sa prevrátili tri kamióny s číslami a všetky čísla sa rozsypali. Keď sa kamióny podarilo postaviť na kolesá, tak vodiči začali nakladať čísla naspäť. Popri tom ale zabudli ako boli rozdelené do kamiónov. Jediné čo vedia je, že *súčin* čísiel v prvom kamióne je záporný, v druhom kladný a v treťom nulový. Taktiež vedia, že ani jeden kamión nebol prázdny. Pomôžte im nájsť správne rozdelenie čísiel.

Úloha

Na vstupe dostanete n čísiel. Vašou úlohou je rozdeliť všetky čísla do troch neprázdnych skupín tak, aby sučin čísiel v prvej skupine bol záporný, v druhej kladný a v tretej nulový.

Čísla na vstupe sa môžu opakovať, nie sú zaručene rôzne. Zachovajte počet, aby bola každá hodnota na výstupe dokopy presne toľkokrát, ako na vstupe.

Je zaručené, že takéto rozdelenie je možné.

Formát vstupu

Na prvom riadku vstupu je číslo n ($3 \leq n \leq 10^5$).

Na nasledujúcom riadku je n medzerou oddelených celých čísiel a_i.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4
$3 \leq n \leq$	10	100	1 000	10^5
$\max \lvert a_i \rvert \leq$	10	100	1 000	10^5

Formát výstupu

Vypíšte tri riadky. Prvý riadok obsahuje čísla, ktorých súčin je záporný, druhý riadok čísla, ktorých súčin je kladný. Na poslednom riadku sú čísla s nulovým súčinom. Každý riadok má nasledujúci formát: Prvé číslo určuje, koľko čísiel je v danej skupine. Následujú čísla, ktoré sú v danej skupine. Medzi číslami je vždy práve jedna medzera. Za posledným číslom medzera nie je. Čísla v jednotlivých skupinách môžu byť v ľubovoľnom poradí.

Príklady

vstup

```
3
1 0 -1
```

výstup

```
1 -1
1 1
1 0
```

vstup

```
4
-1 0 1 2
```

výstup

```
1 -1
1 2
2 0 1
```

Ďalšia správna odpoveď je aj ak je v druhej skupine iba číslo 1 *a v tretej skupine čísla* 2 *a* 0.

3. Aspoň jedna cesta von

12 b za popis, 8 b za program

Po veľkom úspechu pri stavbe druhého slovenského mrakodrapu v Kokave nad Rimavicou sa dcérska firma Trojstenu (ktorá sa zaoberá cestnými stavbami) rozhodla pustiť do stavby ciest. Keďže zamestnanci v tejto firme sú výrazne lepší programátori ako robotníci, tak sa nejaké cesty síce postavili, ale iba tak veľmi náhodne. Čo je ešte horšie, všetky postavené cesty sú iba jednosmerné, na druhý smer sa trochu zabudlo.

To sa ľuďom ani trochu nepáčilo a teda sa začali búriť. Aby vedenie Trojstenu upokojilo situáciu, rozhodlo sa sľúbiť ľuďom, že označí postavené cesty tak, aby z každého mesta viedla aspoň jedna cesta. Bohužiaľ, tento nerozvážny sľub vedenie dalo pred tým, než si overili, že to je možné.

Keďže všetci v Trojstene sa vybrali otáčať cedule na cestách, tak neostal nikto, kto by vlastne zistil, či je ich možné pootáčať tak aby splnili sľub. Vedeli by ste to zistiť vy?

Úloha

Máme cestnú sieť, ktorá sa skladá z miest a ciest medzi nimi. Tieto cesty sú všetky jednosmerné a vašou úlohou je zistiť, či ich vieme orientovať tak, aby z každého mesta išla von aspoň jedna cesta. Každá cesta spája vždy práve dve mestá, nemôže ísť z mesta do toho istého mesta a medzi dvomi mestami vedie vždy najviac jedna cesta.

Formát vstupu

V prvom riadku vstupu sú dve medzerou oddelené čísla n, m ($1 \leq n \leq 10^6$, $0 \leq m \leq 10^6$) počet miest a počet ciest.

Nasleduje m riadkov, na každom sú dve medzerou oddelené čísla a, b ($0 \leq a, b < n, a \neq b$), ktoré hovoria, že mestá a a b sú spojené cestou. Mestá sú označené číslami od 0 po $n - 1$.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4
$1 \leq n \leq$	20	100	1 000	100 000
$1 \leq m \leq$	40	10 000	1 000 000	1 000 000

Formát výstupu

Vypíšte jediný riadok, na ktorom je buď text `ano`, ak vieme orientovať cesty tak, aby z každého mesta viedla aspoň jedna cesta. Ak to nejde, vypíšte `nie`.

Príklady

vstup

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

výstup

```
ano
```

V tomto prípade vieme napríklad orientovať cesty tak, že cesty idú `1 -> 0`, `0 -> 2` a `2 -> 0`.

vstup

```
4 3
0 1
1 2
2 0
```

výstup

```
nie
```

Podobný prípad, ale máme o jedno mesto naviac (mesto 3). Z tohoto mesta žiadna cesta vychádzať nemôže, keďže na žiadnej ceste nie je.

vstup

```
4 2
0 1
2 3
```

výstup

```
nie
```

Nech orientujeme cesty akokoľvek, nevieme zariadiť, aby zo všetkých miest vychádzala aspoň jedna cesta.

vstup

```
6 6
0 1
1 2
2 0
3 4
4 5
5 3
```

výstup

```
ano
```

Ak cesty orientujeme ako `0 -> 1`, `1 -> 2`, `2 -> 0`, `3 -> 4`, `4 -> 5`, `5 -> 3`, tak z každého mesta vychádza aspoň jedna cesta.

4. Len tak ďalej Adam

12 b za popis, 8 b za program

Adama už nebaví, ako si z neho všetci robia srandu, že je malý. Preto sa rozhodol to zmeniť – začne posilovať. Tak sa jedného dňa dotrepal do posilovne. Už už išiel robiť prvý set, keď sa pri ňom zastavil tréner: "Servus mladý! Vieš, ako sa to tu robí?"

Adam mu vraví: "Dobrý! Nie, som tu po prvýkrát."

Tak sa tréner pustil do vysvetľovania: "U nás máme špeciálny cvičiaci plán. Aby si mal dobrý workout, odcvičíš so všetkými činkami, ktoré máme. Ale musíš cvičiť v špeciálnom poradí. Každá činka je zavesená na nejakej inej činke, teda okrem poslednej, tá visí na stojane. Vždy, keď ideš cvičiť, tak si musíš zobrať činku, na ktorej nie sú zavesené žiadne iné. Keď máš na výber, musíš zobrať najľahšiu takú. Keď s činkou docvičíš, nevracaj ju tam kde visela, proste ju polož na kopu na zem. A nezabudni si zapisovať, v akom poradí si cvičil, aby som to mohol skontrolovať."

Adam je však známy tým, že je veľmi zábudlivý, takže si vždy spomenul na to, že si to má zapisovať, až keď držal činku v ruke. Keďže sa mu nechcelo zdvihnúť činku ku očiam, aby dovidel na číslo hmotnosti, tak si zakaždým zapísal na lepiaci papierik iba váhu tej činky, z ktorej tú svoju odvesil. Potom tento papierik nalepil na stenu. Aby ušetril miesto na stene, ďalší papierik nalepil vždy na ten predchádzajúci.

Keď nakoniec prišiel tréner a uvidel Adamove zápisky, vôbec nebol spokojný! Chce od neho, aby mu povedal, v akom poradí s činkami cvičil. Chudák Adam teda postupne odlepuje papieriky zo steny, a snaží sa podľa toho zistiť v akom poradí činky používal. Aby upokojil netrpezlivého trénera, potrebuje to rýchlo, teda musí po každom papieriku povedať dalšiu činku, ktorú použil.

Úloha

Máme n činiek s váhami $1, \ldots, n$. Na začiatku boli usporiadané tak, že jedna bola na stojane, a všetky ostatné viseli na nejakej inej činke. Činky môžu byť zavesené bez ohľadu na váhu, aj ľahšia na ťažšej, aj ťažšia na ľahšej.

Keď Adam ide cvičiť s činkou, dodržiava nasledovné:

1) môže cvičiť len s činkami, na ktorých nie je nič zavesené
2) vždy cvičí s najľahšou dostupnou činkou

Zakaždým si na nový papierik napíše číslo činky, z ktorej túto činku zvesil, a ten potom nalepí na predchádzajúci papierik na stene.

Ako poslednú činku zoberie tú, čo visí na stojane. Keďže nevisí na inej činke, o nej si nezapíše žiaden papierik.

Teraz sú papieriky nalepené na stene – v opačnom poradí ako s nimi Adam cvičil. Adam teraz musí povedať v akom poradí s činkami cvičil od poslednej po prvú, a to tak, že postupne odliepa všetky papieriky a s každým odlepeným musí povedať ďalšiu aspoň jednu činku z tohoto poradia.

Formát vstupu

V prvom riadku vstupu je čislo n - počet činiek, s ktorými Adam cvičil.

Nasleduje $n-1$ riadkov. Na každom je číslo od 1 po n zodpovedajúce papieriku na stene, teda o každej činke, s ktorou Adam vtedy cvičil, váha tej činky, na ktorej bola zavesená. Riadkov je $n-1$ kvôli tomu, že činka na stojane nemá papierik.

Tieto čísla sú napísané v opačnom poradí v akom s nimi Adam cvičil.

Pre n platia nasledovné obmedzenia:

Sada	1, 5	2, 6	3, 7	4, 8
$1 \leq n \leq$	10	1000	10^5	10^5

Formát výstupu

Vypíšte n riadkov, na i-tom z nich váhu činky, s ktorou Adam cvičil ako i-tou poslednou. Teda najprv vypíšte váhu činky, čo bola na stojane, a ako poslednú vypíšte váhu činky, s ktorou Adam cvičil úplne na začiatku.

V sadách 1 až 4 máte dovolené najprv načítať všetky papieriky, a až potom odpovedať. V sadách 5 až 8 musíte hneď po prečítaní každého papierika vypísať aspoň jeden riadok výstupu.

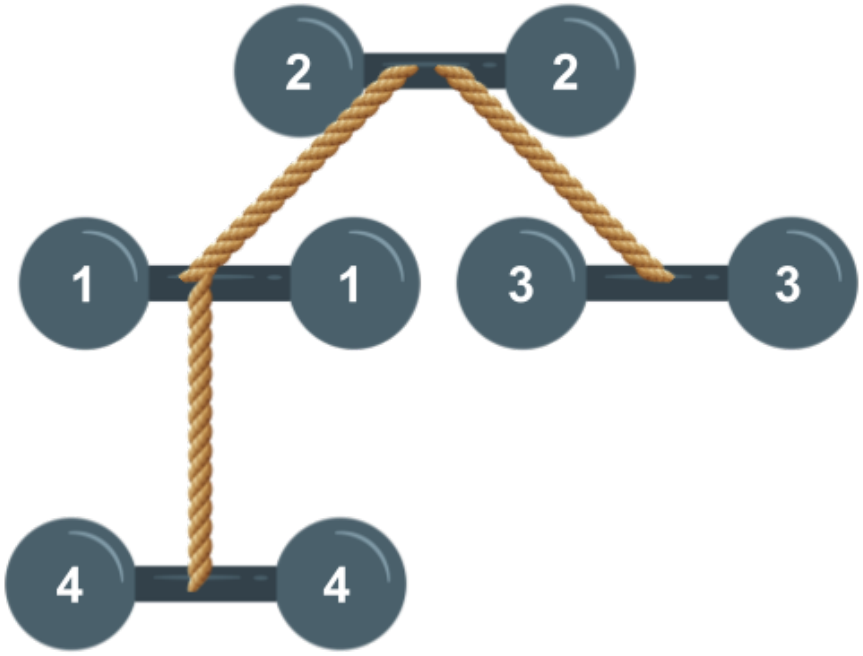
Aby testovanie fungovalo ako má, **je nutné** po vypísaní každého riadku flushnúť váš výstup, aby ho testovač uvidel. V C++ by malo stačiť `cout << nieco << endl;`, ale môžete explicitne použiť príkaz `cout.flush();`. V Pythone vypisujte príkazom `print(cislo, flush=True)` alebo po vypísaní spravte `sys.stdout.flush()`. Pre iné jazyky hľadajte ekvivalent k príkazu `flush`.

Príklady

Takýto vstup môže byť v 5. sade:

```
>>> 4   #počet činiek
>>> 2   #prvý papier
<<< 2   #posledná činka mala váhu 2
>>> 1
<<< 1
>>> 2
<<< 4
<<< 3   #na konci vypíš uplne prvú činku
```

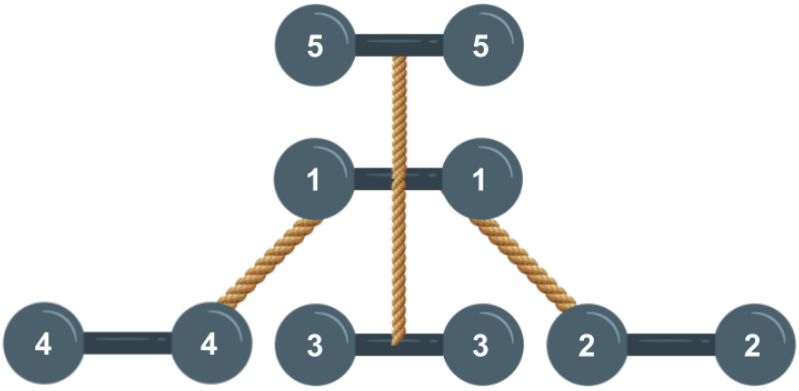
v tomto prípade na začiatku boli činky odložené takto:



Adam cvičil s činkami v poradí 3, 4, 1, 2. Na papieriky postupne napísal 2, 1, 2.
Takýto vstup by mohol byť v 1. sade:

```
>>> 5   #počet činiek
>>> 5   #činky
>>> 1
>>> 1
>>> 1
<<< 5   #posledná činka
<<< 1
<<< 4
<<< 3
<<< 2   #prvá činka
```

v tomto prípade na začiatku boli činky odložené takto:



Adam cvičil s činkami v poradí 2, 3, 4, 1, 5. Na papieriky postupne napísal 5, 1, 1, 1.

5. Neskutočná zápcha

12 b za popis, 8 b za program

Na našej diaľnici pravidelne vzniká hrozná zápcha. Výbor pre modernizáciu diaľníc sa rozhodol konečne s tým niečo robiť. Zavolali si rôznych školených expertov, aby im s tým poradili. Prvý odborník povedal, že treba spraviť kruhový objazd, všetci predsa vedia, že proti zápche pomáhajú. Druhý odborník povedal, že zápcha by sa vyriešila, keby ľudia namiesto osobných áut cestovali autobusom. Tretí odborník povedal, že v Japonsku majú špičkový turbointeligentný semafór, čo má vstavanú AIčku na reguláciu zápchy.

Keď sa odborníci večnosť nevedeli dohodnúť, koho spôsob je najlepší, výbor sa nakoniec rozhodol proste skombinovať všetky ich návrhy. Postavili obrovský kruháč a pri výjazde z neho namontovali japonský turbosemafór. Tento honosný výtvor nazvali prvou turbookružnou križovatkou. Ale stala sa osudná chyba. Zabudli, že v Japonsku sa jazdí po ľavej strane, takže semafór je naopak. Namiesto toho, aby púšťal autobusy z kruháča von, sú nútení ďalej krúžiť. Zápcha je tisíckrát horšia, ako predtým.

Úloha

Na turbookružnej križovatke stojí v rade pred semafórom n autobusov. Turbosemafór pomocou AIčky rozoznáva, koľko majú cestujúcich. Na každú zelenú pustí niekoľko autobusov, a to tak, aby počet ľudí neprekročil r. Tie budú pokračovať v krúžení a zaradia sa naspäť na koniec radu v rovnakom poradí. Prvý autobus, ktorý by spôsobil že súčet stúpne nad r, dostane červenú a musí čakať.

Ako špeciálny prípad, žiaden autobus nemôže prejsť viackrát počas tej istej zelenej. Aj keby bolo r dostatočne veľké, ak v rámci jednej zelenej semafór pustí všetky autobusy, prepne sa okamžite na červenú.

Cestujúcim ešte zostáva posledný kúsok nádeje. Po k zelených to turbosemafór nezvládne, prehreje sa mu grafická karta a exploduje. Potom budú môcť všetky autobusy konečne odísť.

Výbor by chcel odhadnúť, koľko ekonomickej škody a stratenej produktivity to celé spôsobilo. Preto potrebujú zistiť súčet, koľko ľudí prešlo križovatkou na každú zelenú až kým sa semafór nepokazil. S touto úlohou si však nevedia dať rady, preto pripadla ako obvykle vám.

Formát vstupu

Na prvom riadku vstupu sú tri medzerou oddelené čísla r, k, n ($1 \leq r, k \leq 10^9, 1 \leq n \leq 10^6$) kde r je maximálny počet ľudí čo môže prejsť na jednu zelenú, k je počet zelených, a n je počet čakajúcich autobusov.

Na nasledujúcom riadku je n medzerou oddelených čísel $a_1, \ldots, a_n$ ($1 \leq a_i \leq r$), i-te z nich popisuje počet ľudí v i-tom autobuse. Prvý autobus stojí na začiatku radu hneď pri semafóre, n-tý na konci.

Formát výstupu

Vypíš jeden riadok a v ňom jedno celé číslo, počet ľudí, ktorí prejdú turbookružnou križovatkou.

Hodnotenie

Sú štyri sady testovacích vstupov. V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4
$1 \leq n \leq$	1 000	10^4	10^6	10^6
$1 \leq k, r \leq$	1 000	10^4	10^9	10^9
Dodatočné obmedzenia	-	-	$\forall i, j : a_i = a_j$	-

Pozor! Odpovede sa nemusia zmestiť do bežných celočíselných premenných. Preto odporúčame použiť napríklad premenné typu `long long` *v* `c++`*.*

Príklady

vstup

```
11 6 5
3 7 8 8 8
```

výstup

```
52
```

Na prvú zelenú semafór pustí prvý a druhý autobus (spolu 10 ľudí). Keď sa zaradia na koniec, nové poradie bude [8, 8, 8, 3, 7]. Na druhú zelenú prejde jeden autobus s 8 ľuďmi a na tretiu ďalší 8. V tomto momente to bude [8, 3, 7, 8, 8]. Na štvrtú zelenú prejdú autobusy 8 (čo bol pôvodne posledný) a 3 (pôvodne prvý). Na piatu zelenú prejde 7 a na šiestu 8. Spolu prejde $10 + 8 + 8 + 11 + 7 + 8 = 52$ *ľudí.*

vstup

```
99 3 5
1 2 3 4 5
```

výstup

```
45
```

Na každú zelenú prejde všetkých 5 autobusov. Semafór by púšťal aj viac ľudí, ale už nemá koho.

6. Ilúzia luxusu

12 b za popis, 8 b za program

Keď sa chcú normálni ľudia vyvyšovať nad svoje okolie a dať viditeľne najavo svoje bohatstvo alebo status, kúpia si športové auto, švajčiarske hodinky, alebo značkovú koženú kabelku. Ale čo majú robiť chudáci kockáči? Ako majú ukázať svojim známym že nie sú nijaký plebs, aby to pochopil aj ten najasociálnejší introvert?

Podnikaví KSPáci si založili firmu, ktorá je zameraná na presne tento trhový segment. Prišli na trh s novým produktom: luxusnými organickými šachovnicami. Každý kockáč, čo ju uvidí, zaručene zbledne závisťou!

Organické šachovnice sa vyrábajú z kôry *šachovnicovníka balkánskeho*, národného stromu Chorvátska. Táto kôra má veľmi charakteristické zafarbenie: v pravidelnej mriežke na nej prirodzene vznikajú krásne biele a čierne štvorčeky. Hotový zázrak prírody.

Priviezli sme z Chorvátska veľký obdĺžnik odrezanej kôry. Musíme ho nakrájať na šachovnice. Môžu byť rôzne veľké, ale čím väčšie tým lepšie. Keďže šachovnicovníky sú veľmi vzácne a treba ich kôru dovážať z takej diaľky, chceli by sme ju celú spotrebovať až do posledného štvorčeka.

Úloha

Máme nejaký obdĺžnik zložený z bielych a čiernych štvorčekov. Pri výrobe použijeme tento proces:

Nájdeme v obdĺžniku najväčšiu šachovnicu, ktorú z neho môžeme vyrezať. Šachovnica je definovaná ako ľubovoľne veľký štvorec, v ktorom sa nikde stranou nedotýkajú políčka rovnakej farby. Ak máme viaceré rovnako veľké možnosti, vyberieme najvrchnejšiu, a ak sú stále viaceré v rovnakej výške, najľavejšiu. Danú šachovnicu vyrežeme a môžeme predávať.

Toto opakujeme až kým nevyrežeme úplne každé políčko. Ku koncu, keď sa všetky väčšie minú, možno bude treba vyrezávať aj 1x1 šachovnice (jednotlivé štvorčeky). Naše marketingové oddelenie si s tým nejak poradí.

Zistite, koľko šachovníc ktorej veľkosti vznikne touto stratégiou.

Formát vstupu

V prvom riadku vstupu je výška obdĺžnika r a šírka obdĺžnika c.

Zvyšok vstupu tvorí r riadkov, každý popisuje jeden riadok nášho čiernobieleho obdĺžnika.

Aby sme ušetrili trochu miesta, každý riadok je na vstupe zapísaný ako hexadecimálne číslo[1], zložené z presne $c/4$ cifier (znakov `0-9A-F`). Každá cifra teda udáva farbu štyroch štvorčekov. Čísla sú zapísané normálne,

[1] https://sk.wikipedia.org/wiki/%C5%A0estn%C3%A1stkov%C3%A1_s%C3%BAstava

od veľkého konca (big-endian), čiže najvýznamnejšia cifra je vľavo. Napríklad hexadecimálny zápis `CFF8` je v binárke `1100111111111000` – `C` je `1100` atď. Nulové bity sú čierne štvorčeky a jednotkové bity sú biele.

Šírka c je zaručene deliteľná štyrmi.

V jednotlivých sadách platia nasledujúce obmedzenia:

Sada	1	2	3	4
$1 \leq r, c \leq$	12	64	256	1024

Formát výstupu

Najprv vypíšte riadok s číslom k – počtom rôznych veľkostí vyrobených šachovníc.

Potom vypíšte k riadkov a na každom dve kladné celé čísla: veľkosť (dĺžka strany) šachovnice, a počet kusov vyrobených šachovníc čo má túto veľkosť.

Veľkosti musia byť zoradené zostupne, čiže v takom poradí ako sme ich vyrezávali.

Príklady

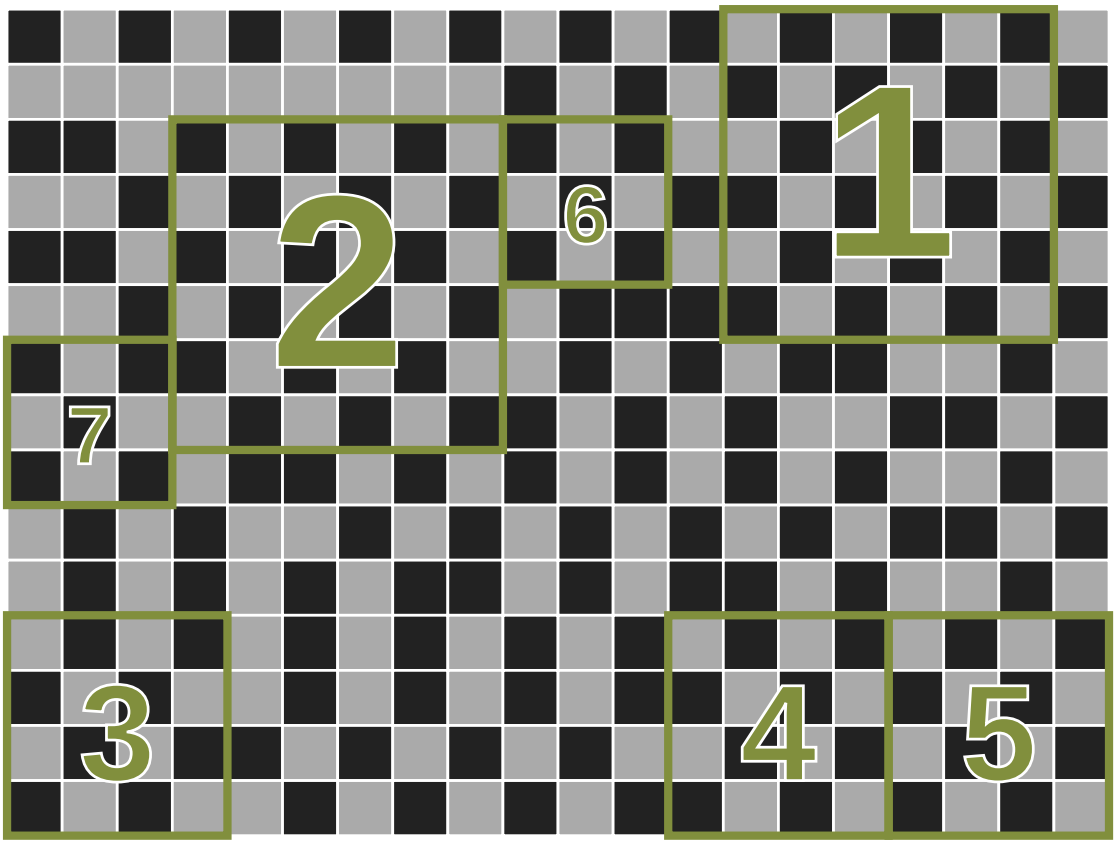
vstup

```
15 20
55555
FFAAA
2AAD5
D552A
2AAD5
D542A
4AD4D
B52B2
52AAD
AD552
AA52D
AAAAA
5AA55
A55AA
5AA55
```

výstup

```
5
6 2
4 3
3 7
2 15
1 57
```

Na obrázku je znázornené poradie prvých 7 šachovníc, ktoré sme našli a vyrezali. Vidno, že prvá a druhá sú veľké 6x6, ďalšie tri sú 4x4, a potom režeme 3x3. Vyrezávanie pokračuje aj ďalej, ďalšími piatimi 3x3 a potom mnohými 2x2 a 1x1, ale tie už na obrázku nakreslené nie sú.



vstup

```
4 4
0
0
0
0
```

výstup

```
1
1 16
```

Aká smola, máme 16 čiernych štvorčekov… Marketingové oddelenie sa bude musieť fakt snažiť.

vstup

```
4 4
3
3
C
C
```

výstup

```
2
2 1
1 12
```

vstup

```
4 8
6A
95
9A
65
```

výstup

```
2
4 1
2 4
```

7. Celistvé tabule

12 b za popis, 8 b za program

Kde bolo, tam bolo, pri rozostavanej diaľnici stála tabuľa “nevyrušujte, tu staviame diaľnice!”. Až jedného dňa vietor zavial a tabuľa sa rozpadla na maličké kúsky. Takže nielen diaľnicu treba stavať, ale aj novú tabuľu.

Lenže robotníci na stavbe si myslia, že pôvodný nápis už začínal byť nudný (napokon, už dlho žiadnu diaľnicu nestavali). Radi by ho nahradili iným humorným nápisom. Ale nanešťastie nemajú dosť farby, aby namaľovali nový nápis. Ani z pôvodnej tabule nezostali vôbec žiadne použiteľné zvyšky.

Našťastie objavili, že medzi stavebnými materiálmi majú pripravené množstvo menších tabúľ, rovno aj s nápismi.

Nanešťastie, nikomu sa žiadny z existujúcich nápisov nepáčil.

Našťastie jeden robotník dostal nápad: "čo keby sme zobrali dve tabule a dali ich vedľa seba, aby vzniklo niečo smiešne?"

Nanešťastie, spájať tabule nie je len tak (veď sme videli, ako ľahko sa tá stará rozpadla). Keby vyrobili nápis len tak prilepením dvoch tabúľ stranou k sebe, za chvíľu by bolo po ňom.

Našťastie dostali ďalší geniálny nápad: nemohli by proste zobrať tretiu tabuľu, prelepiť ju cez dve spojené tabule, a takto dostať celistvú tabuľu?

Nanešťastie, to nemôžete len tak lepiť všakovaté tabule cez tabule, čo keby sa vrchná tabuľa odlepila a ukázalo sa, že zakrývala úplne iný nápis! Treba aby sa nápis na tretej tabuli presne zhodoval so zakrytým textom na prvých dvoch. A samozrejme, tretia tabuľa musí zakrývať časť oboch spodných, aby to držalo pokope.

Tak, všetko je vyriešené, a už len ostáva vymyslieť ktorú trojicu tabúľ použijete na výrobu novej (celistvej) tabule.

Našťastie, na to ste tu vy, aby ste zistili z koľkých možností si môžu vyberať!

Úloha

Máte n kôpok tabúľ označených 1 až n. Na kôpke i je množstvo tabúľ s nápisom w_i. Spočítajte počet možností, koľkými spôsobmi sa dajú správne zlepiť. Správne zlepené tabule vyzerajú napríklad takto:

```
(Ľ) DIAĽNICE..........
(P) ........NASTAVANIE
(S) ......CENA........
```

(Ľ)avá a (P)ravá tabuľa sa musia tesne dotýkať, ale nie prekrývať. (S)tredná tabuľa sa musí aspoň jedným znakom prekrývať s ľavou aj pravou tabuľou. Jej nápis sa musí zhodovať so zakrytým textom, a nesmie prečnievať za ľavý okraj ľavej ani pravý okraj pravej tabule.

- Toto nie je dobrá možnosť, lebo nápis na strednej tabuli sa nezhoduje so zakrytým textom:

  ```
  (Ľ) DIAĽNICE.........
  (P) ........NESTAVAME
  (S) ......CENA.......
  ```

- Toto nie je dobrá možnosť, lebo stredná tabuľa zakrýva iba pravú, takže ľavá môže ľahko odpadnúť:

  ```
  (Ľ) DIAĽNICA......
  (P) ........NEBUDE
  (S) ..........BUDE
  ```

- Toto nie je dobrá možnosť, lebo stredná tabuľa pokračuje za okraj pravej:

  ```
  (Ľ) POCHOPIT......
  (P) ........NAVOD.
  (S) .....PITNAVODA
  ```

Počítanie možností

Každá možnosť je jedinečne identifikovaná štyrmi parametrami: číslom kôpky, z ktorej berieme ľavú tabuľu, číslom kôpky, z ktorej berieme pravú tabuľu, číslom kôpky, z ktorej berieme strednú tabuľu, a miestom, kde priložíme strednú tabuľu cez ľavú a pravú. Ak sa čokoľvek z toho líši, ráta sa to ako ďalšia možnosť. Nejde len o počet rôznych výsledných viditeľných nápisov, ale o všetky rôzne spôsoby, ako ich zložiť.

Ak sú viaceré možné pozície, kde v texte sa dá priložiť nejaká stredná tabuľa na nejakú ľavú a pravú, započítajte každú pozíciu ako ďalšiu možnosť.

Ak majú viaceré kôpky tabúľ rovnaký nápis, započítajte samostatne každú platnú kombináciu čísel kôpok.

Na každej kôpke je veľa kusov tabúľ s tým istým nápisom w_i ("veľa" znamená "aspoň 3 kusy"). Takže viaceré tabule (ľavá, pravá a/alebo stredná) môžu mať ten istý nápis, aj keby bola na vstupe len jedna kôpka s takým nápisom. Inými slovami, aj možnosti, ktoré používajú to isté číslo kôpky na viacerých miestach, sú platné.

Formát vstupu

V prvom riadku vstupu je číslo n ($1 \leq n \leq 2 \cdot 10^5$) udávajúce počet kôpok tabúľ.

Na každom z nasledujúcich n riadkov je jeden reťazec w_i pozostávajúci z veľkých písmen anglickej abecedy – nápis na tabuliach na kôpke i. Nápisy nemusia nutne byť rôzne.

Celkový súčet dĺžiek reťazcov na vstupe nepresiahne 10^6.

Existuje 8 sád vstupov. Majú nasledovné obmedzenia:

Sada	maximálne n	ďalšie obmedzenia
1	50	Dĺžka každého reťazca je najviac 50
2	100	Dĺžka každého reťazca je najviac 100
3	10	Bez obmedzení
4	5000	Súčet dĺžok reťazcov nepresiahne 10^4
5	10^6	Reťazce pozostávajú iba z písmena A
6	10^6	Dĺžka každého reťazca je najviac 30
7	10^6	Dĺžka každého reťazca je najviac 60 a reťazce pozostávajú iba z písmen "A, B"
8	10^6	Bez obmedzení

Formát výstupu

Vypíš jeden riadok a v ňom jedno celé číslo: počet rôznych správnych možností, ako zlepiť tabule zo vstupu.

Príklady

vstup

```
7
DIALNICE
DIALNICA
NASTAVANIE
NESTAVAME
NEBUDE
BUDE
CENA
```

výstup

```
1
```

Jediná možnosť je hore uvedená trojica tabúľ (`DIALNICE`, `NASTAVANIE`, `CENA`) so znázornenou pozíciou.

vstup

```
3
A
BABA
ABAB
```

výstup

```
8
```

Nápis `A` je príliš krátky aby išiel použiť ako stredná tabuľa, ale môžeme ho použiť ako ľavú alebo pravú tabuľu a úplne ho zakryť. Všimnite si, že existujú viaceré možnosti ktoré vytvoria rovnaký nápis, napr. pri L=`BABA` P=`BABA` S=`ABAB` môžeme nalepiť strednú tabuľu na pozíciu doľava na `B[ABAB]ABA` alebo doprava na `BAB[ABAB]A`. Tiež si všimnite, že sme v tomto prípade zobrali ľavú aj pravú tabuľu z tej istej kôpky (`BABA`). Dokonca existujú aj možnosti, kde zoberieme všetky tri tabule z tej istej kôpky.

vstup

```
2
HAHA
HAHA
```

výstup

```
8
```

Môžeme si vybrať, či ľavú tabuľu vezmeme z kôpky 1 alebo kôpky 2, pravú z 1 alebo 2, a strednú z 1 alebo 2, čo sa technicky ráta ako $2^3 = 8$ rôznych možností. V každom prípade vznikne nápis `HAHAHAHA`, a strednú tabuľu musíme umiestniť tak, aby zakrývala to stredné `HAHA`.

8. Akútna výstavba

12 b za popis, 8 b za program

Koho by už len zaujímala diaľnica, aj tak všetci vieme, že to hlavné, čo sa v okolí Kokavy nad Rimavicou bude stavať je predsa masivácky mrakodrap. Poďme sa teda za ľubozvučných zvukov výstavby pozrieť na aktuálnu situáciu na stavenisku.

Asi si viete presne predstaviť, ako to na takej stavbe vyzerá. Väčšina robotníkov len tak postáva okolo a len zopár vyvolených pracuje. A aby to nebolo málo, tak ešte chodia aj na pravidelné prestávky. Stavbyvedúceho Dušana už samozrejme nebaví toto všetko sledovať. To ale nemení nič na tom, že stále potrebuje svojich nadriadených informovať o postupe stavby. Na to ale potrebuje vždy vedieť, koľko chlapov aktuálne pracuje.

Presne preto povolal vás, lacnopracujúcich brigádnikov, aby ste mu tieto dáta pomohli získať a on si naďalej mohol v kľude užívať túto malebnú scenériu.

Úloha

Na vstupe dostanete popis jednotlivých Dušanových robotníkov. Kým je i-ty robotník prítomný na stavenisku, najprv prvých a_i hodín obdivuje prácu ostatných, a následne ďalších b_i hodín maká. Potom zase a_i hodín obdivuje a b_i hodín maká, a takto sa to pravidelne strieda, kým zas neodíde zo staveniska preč dať si pauzu.

Každú hodinu nastane jedna z dvoch situácií. Buď nejaký robotník príde z pauzy na stavenisko (a začne hneď svoj pracovný cyklus tým, že bude obdivovať ostatných) alebo sa naopak jeden robotník na stavenisku rozhodne, že už toho má dosť, a ide pauzovať.

Dušana by po každej takejto zmene zaujímalo, koľko robotníkov počas nasledujúcej hodiny na výstavbe aktívne pracuje.

Na začiatku sú všetci robotníci na pauze.

Formát vstupu

Na prvom riadku vstupu sa nachádzajú čísla n a m ($1 \leq n, m \leq 2 \cdot 10^5$)[2], reprezentujúce počet robotníkov a počet hodín, ktoré prebieha stavba.

Nasleduje n riadkov, na každom z nich sú dve čísla a_i a b_i ($1 \leq a_i, b_i \leq 2 \cdot 10^5$), ktoré popisujú pracovný cyklus i-teho robotníka. Tento robotník najprv a_i hodín obdivuje prácu ostatných a potom b_i hodín aktívne pracuje. Toto sa opakuje, až kým znova nepôjde pauzovať.

Na ďalších m riadkoch sa nachádzajú dve čísla op a k ($0 \leq k < n$), ktoré hovoria o tom, čo sa v danú hodinu stalo. Ak $op = 1$, tak sa robotník číslo k vrátil z pauzy. Ak $op = -1$, tak naopak robotník číslo k práve odchádza na pauzu. Ak má nejaký robotník prísť z pauzy, tak doteraz určite bol na pauze a ak má naopak odísť na pauzu, tak doteraz určite bol na stavenisku. Robotníkov číslujeme od 0.

V jednotlivých sadách navyše platia nasledujúce obmedzenia:

Sada	1	2	3, 4
$1 \leq n, m \leq$	10 000	200 000	200 000
$1 \leq a_i, b_i \leq$	250	250	200 000

Formát výstupu

Pre každú hodinu (teda tesne po tom, čo nejaký robotník odíde alebo príde) vypíšte na samostatný riadok jedno číslo, počet robotníkov, ktorí v túto hodinu aktívne pracujú.

Upozornenie

Odporúčame použiť rýchle načítavanie vstupu v jazyku `C++`. Teda namiesto `std::endl` použiť `\n` a vypnúť sychronizáciu s C-čkovým IO pomocou `cin.tie(0)->sync_with_stdio(0)`.

[2] Vedúcemu odborov Fipovi by sa takéto neľudsky dlhé pracovné hodiny asi nepáčili, ale mrakodrap treba dostavať čo najrýchlejšie…

Príklad

vstup

```
3 6
1 1
6 3
2 1
1 0
-1 0
1 0
1 2
1 1
-1 1
```

výstup

```
0
0
0
1
0
2
```

- *Prvá hodina začne príchodom robotníka 0, ktorý ju strávi obdivovaním rozostavaného mrakodrapu a osamelou meditáciou o význame stavby pre rozvoj a blahobyt dediny.*
- *V druhej hodine bude po jeho odchode stavenisko opustené, teda aj teraz bude aktívne pracovať 0 robotníkov.*
- *V tretej hodine sa vráti, ale aj keď už predtým hodinu obdivoval, začne zase od začiatku obdivovaním.*
- *Vo štvrtej hodine konečne prvýkrát začne robotník 0 aktívne pracovať a robotník 2 obdivuje, ako mu to ide.*
- *V piatej hodine ale znova prestane a robotníci 1 a 2 tiež len obdivujú, čo všetko už stihol postaviť.*
- *Nakoniec v šiestej hodine budú aktívne pracovať robotníci 0 aj 2. Aký úspešný deň!*