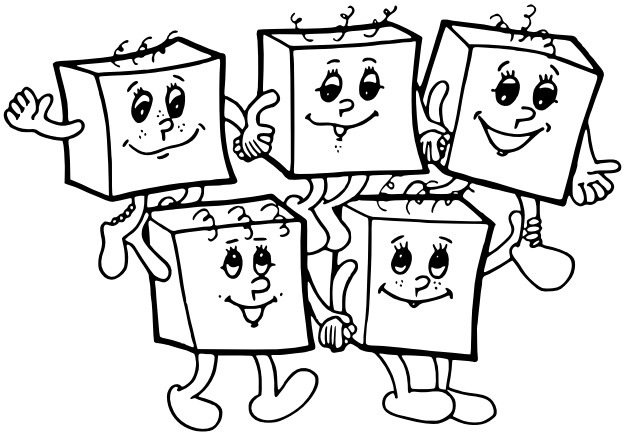
OLYMPIÁDA V INFORMATIKE

NA STREDNÝCH ŠKOLÁCH

http://oi.sk/



dvadsiaty piaty ročník
školský rok 2009/10

zadania celoštátneho kola
kategória A

1. súťažný deň

Priebeh celoštátneho kola

Celoštátne kolo 25. ročníka Olympiády v informatike, kategórie A, sa koná v dňoch 24.-27. marca 2010. Na riešenie úloh prvého, teoretického dňa majú súťažiaci 4,5 hodiny čistého času. Rôzne úlohy riešia súťažiaci na samostatné listy papiera. Akékoľvek pomôcky okrem písacích potrieb (napr. knihy, výpisy programov, kalkulačky) sú zakázané.

Čo má obsahovať riešenie úlohy?

- Slovne popíšte algoritmus.
  Slovný popis riešenia musí byť jasný a zrozumiteľný i bez nahliadnutia do samotného algoritmu/programu.
- Zdôvodnite správnosť vášho algoritmu.
- Uveďte a zdôvodnite jeho časovú a pamäťovú zložitosť.
- Podrobne uveďte dôležité časti algoritmu, ideálne vo forme programu v Pascale alebo C/C++.
- V prípade, že používate vo svojom programovacom jazyku knižnice, ktoré obsahujú implementované dátové štruktúry a algoritmy (napr. STL pre C++), v popise algoritmu stručne vysvetlite, ako by ste napísali program s rovnakou časovou zložitosťou bez použitia knižnice.

Hodnotenie riešení prvého (teoretického) dňa

Za každú úlohu môžete získať od 0 do 10 bodov.

Pokiaľ nie je v zadaní povedané ináč, základným kritériom hodnotenia riešenia je **správnosť** a **efektívnosť** navrhnutého algoritmu. Hodnotí sa aj kvalita popisu riešenia a zdôvodnenie tvrdení o jeho správnosti a efektívnosti.

Efektívnosť algoritmu posudzujeme vypočítaním jeho časovej zložitosti – funkcie, ktorá hovorí, ako dlho vykonanie algoritmu trvá v závislosti od veľkosti vstupných parametrov. Spomedzi dvoch algoritmov s rádovo rovnakou časovou zložitosťou je lepší ten, ktorý potrebuje rádovo menej pamäte.

V zadaní úlohy môžu byť uvedené limity na veľkosť premenných. Tieto môžete použiť na odhad toho, ako dobré vaše riešenie je. Na počítači, ktorý vykoná miliardu inštrukcií za sekundu, zrieši vzorové riešenie ľubovoľný povolený vstup do niekoľko sekúnd.

A-III-1 Čokoláda je tu zas

Janko bude mať čoskoro narodeniny. Jeho sestra Marienka si ešte dobre pamätá na olámanú čokoládu, ktorú jej daroval pred pár mesiacmi. Rozhodla sa teda, že sa mu oplatí rovnakou mincou.

Zašla do pivnice a zo svojej tajnej skrýše vybrala čokoládu, ktorú si tam kedysi ukryla. Všadeprítomné myši už aj túto čokoládu ohrýzli, to však Marienke neprekážalo – veď predsa stačí vyhryzené časti olámať.

Marienka je však šikovnejšia ako Janko a uvedomila si, že nemusí lámaním vyrobiť štvorec. Čokolády sú predsa často obdĺžnikového tvaru. A to jej ponúka množstvo nových možností ako vyrobiť darček pre Janka.

Súťažná úloha

Daný je pôvodný počet riadkov R a stĺpcov S, ktoré čokoláda kedysi mala, a matica $R \times S$ núl a jednotiek udávajúca, ktoré políčka z nej zostali celé.

Zistite, koľkými spôsobmi môže Marienka uskutočniť svoj plán. Inými slovami, spočítajte, koľkými spôsobmi sa dá na zvyšku čokolády vyznačiť obdĺžnik bez dier. Všetky hrany obdĺžnika musia samozrejme ležať na hranách políčok. Rovnako veľké obdĺžniky na rôznych súradniciach považujeme za rôzne.

Formát vstupu

V prvom riadku vstupu sú dve medzerami oddelené celé čísla R a S. Nasleduje R riadkov, v r-tom z nich je S medzerami oddelených celých čísel $a_{r,1}, \ldots, a_{r,s}$. Ak je políčko (r, s) celé, je $a_{r,s} = 1$, inak $a_{r,s} = 0$.

Formát výstupu

Vypíšte jeden riadok a v ňom jedno celé číslo – hľadaný počet obdĺžnikov.

Príklad

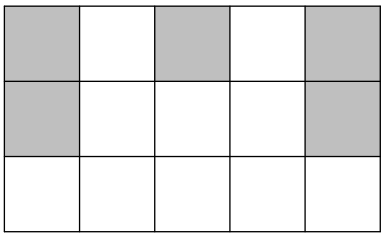
vstup

```
3 5
0 1 0 1 0
0 1 1 1 0
1 1 1 1 1
```

výstup

```
33
```

Na obrázku vpravo je čokoláda popísaná vstupom. Sivou farbou sú políčka, ktoré už myši stihli poškodiť.



Obdĺžnik 1×1 sa na nej dá vyznačiť desiatimi spôsobmi, 2×1 piatimi, 1×2 šiestimi, 3×1 dvoma, 1×3 štyrmi, 1×4 dvoma, 2×2 dvoma, 1×5 jedným, a 2×3 tiež jedným spôsobom.

A-III-2 Odveta

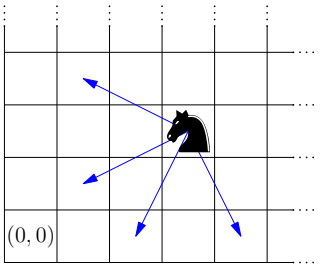
„Šach-mat," s posmešným úškrnom zahlásil CéDéčko. Jeho protihráč Petržlen mal síce doteraz šach veľmi rád, ale už ho to prestáva baviť: práve s CéDéčkom prehral sedemnástu partiu za sebou. Preto si vymyslel novú, vlastnú hru, v ktorej ho určite porazí.

Hracím plánom je šachovnica, ktorá je doprava aj dohora nekonečná. Každé políčko tejto šachovnice teda vieme popísať dvomi nezápornými celými číslami.

Na tejto šachovnici je umiestnených N koní. Na začiatku aj hocikedy počas hry sa môže na jednom políčku naraz nachádzať ľubovoľne veľa koní.

Koňmi je povolené ťahať len podľa šachových pravidiel. Sú povolené len tie ťahy, pri ktorých kôň neopustí šachovnicu a priblíži sa do ľavého dolného rohu. Presnejšie, ťah z (x, y) na (x', y') je povolený len vtedy, keď $x', y' \geq 0$ a $x' + y' < x + y$.

Hráč, ktorý je na ťahu, si vyberie niekoľko koní (môže koľko chce, ale musí aspoň jedného) a každým z nich raz pohne. Hráči ťahajú striedavo. Prehráva ten, pre koho už neexistuje ďalší ťah (teda už nemôže pohnúť žiadnym koňom).



Všetky povolené ťahy koňom na súradniciach $(3, 2)$.

Petržlen si pre túto hru už napísal program, ktorý bude za neho hrať optimálne. Žiaľ, CéDéčko programovať nevie, preto by privítal vašu pomoc.

Súťažná úloha

Je daný počet koňov N a ich umiestnenie na šachovnici na začiatku. Prvý je na ťahu CéDéčko.

Napíšte program, ktorý zistí, kto vyhrá, ak budú obaja hráči hrať optimálne. Ak má byť víťazom CéDéčko, váš program by mu mal poradiť prvý ťah ľubovoľnej vyhrávajúcej stratégie.

V prípade, že úlohu neviete riešiť pre všeobecné N, tak dostanete pár bodov aj vtedy, ak úlohu vyriešite pre jedného koníka, prípadne pre dva koníky začínajúce na súradniciach medzi $(0, 0)$ a $(100, 100)$.

Formát vstupu

Prvý riadok vstupu obsahuje počet koní N. V každom z nasledujúcich N riadkov sú dve čísla r_i a s_i, určujúce riadok a stĺpec, v ktorom sa nachádza i-ty kôň na začiatku hry.

Formát výstupu

Prvý riadok výstupu má obsahovať meno hráča, ktorý vyhrá (`CeDecko` alebo `Petrzlen`). Ak vyhrá CéDéčko, vypíšte pre každého koňa, ktorým má tiahnuť, riadok s číslami r_a, s_a, r_b, s_b – pôvodné a nové umiestnenie koňa. Na poradí týchto riadkov nezáleží.

Príklady

vstup

```
1
0 4
```

výstup

```
Petrzlen
```

CéDéčko musí potiahnuť na $(1, 2)$, odtiaľ Petržlen potiahne na $(0, 0)$ a vyhrá.

vstup

```
2
3 1
2 1
```

výstup

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

Po tomto ťahu CéDéčka Petržlen rovno prehral, lebo ani jedným koňom už nevie pohnúť.
Všimnite si, že keby CéDéčko koňa z $(3, 1)$ nechal na mieste, prehrá.
Tiež by prehral, ak by tohto koňa presunul na $(1, 2)$, bez ohľadu na to, či by druhým koňom pohol alebo nie.

A-III-3 Počítač s gumenou rúrou

V tomto ročníku olympiády sa v teoretickej úlohe stretávame so špeciálnym počítačom zvaným Kvak. V študijnom texte uvedenom za zadaním tejto úlohy je popísané, ako tento počítač funguje. Študijný text je až na štylistické zmeny identický s tým z domáceho kola.

Súťažná úloha

a) (3 body) V rúre je postupnosť kladných celých čísel. Označme ich $a_1, a_2, \ldots, a_N$ v poradí, v akom sa v nej teraz nachádzajú. Napíšte program, ktorý skontroluje, či je táto postupnosť rastúca. Ak áno, mal by váš program skončiť bez toho, aby čokoľvek vypísal. Ak nie je rastúca, mal by nájsť a vypísať najmenšie i také, že $a_i \geq a_{i+1}$.

b) (7 bodov) V rúre je postupnosť kladných celých čísel, pričom viete, že jedno z týchto čísel má v rúre nadpolovičnú väčšinu – toto číslo sa teda v rúre vyskytuje viackrát ako všetky ostatné čísla dokopy. Napíšte program, ktorý toto číslo nájde a vypíše.

Tabuľka inštrukcií

príkaz	účinok príkazu
`get X`	Kvak vyberie číslo z rúry a uloží ho do registra `X`.
`put X`	Kvak vloží do rúry číslo z registra `X`.
`put` číslo	Kvak vloží dané číslo do rúry.
`print`	Kvak vyberie číslo z rúry a vypíše ho na výstup.
`add`	sčítanie: Kvak vyberie dve čísla z rúry a vloží tam ich súčet.
`sub`	odčítanie: Kvak vyberie dve čísla z rúry a vloží tam ich rozdiel (prvé mínus druhé).
`mul`	násobenie: Kvak vyberie dve čísla z rúry a vloží tam ich súčin.
`div`	delenie: Kvak vyberie dve čísla z rúry a vloží tam celú časť ich podielu (prvé lomeno druhé).
`mod`	zvyšok: Kvak vyberie dve čísla z rúry a vloží tam zvyšok, ktorý dá prvé z nich po delení druhým.
`label L`	návestie: Toto miesto v programe dostane meno `L` (kde `L` môže byť ľubovoľný reťazec).
`jump L`	skok: Kvak bude pokračovať vo vykonávaní programu od miesta, ktoré sa volá `L`.
`jz X L`	skok ak nula: Ak je v registri `X` nula, Kvak vykoná príkaz `jump L`.
`jeq X Y L`	skok ak sa rovnajú: Ak je v registroch `X` a `Y` to isté, Kvak vykoná príkaz `jump L`.
`jgt X Y L`	skok ak je väčšie: Ak je v registri `X` väčšia hodnota ako v `Y`, Kvak vykoná príkaz `jump L`.
`jempty L`	skok ak je rúra prázdna: Ak v rúre nie sú žiadne čísla, Kvak vykoná príkaz `jump L`.
`stop`	koniec: Kvak prestane vykonávať program.

Študijný text

Vedci z Kolégia Skúmania Potrubí (KSP) nedávno vyvinuli nový počítač, zvaný Kvak. Kvak má jedinú dátovú štruktúru: jednosmernú gumenú rúru.

Jediný dátový typ, ktorý Kvak pozná, sa volá `number`, a je to celé číslo z rozsahu od 0 po 65 535, vrátane.[1] Všetky matematické operácie počíta Kvak modulo 65 536. Teda napríklad hodnota výrazu 65530 + 10 je 4.

Kvak má 26 premenných, ktoré voláme registre. Tie sú označené písmenami `a` až `z` a v každom z nich môže byť uložená jedna hodnota typu `number`. Na začiatku výpočtu sú vo všetkých registroch nuly.

Okrem registrov má Kvak, ako sme už spomínali, jednu jednosmernú gumenú rúru. S tou vie robiť dve operácie: vložiť do nej číslo z registra `X` príkazom `put X` a z opačného konca rúry vybrať číslo a uložiť ho do registra `X` príkazom `get X`.

[1] $65\,535 = 2^{16} - 1$, typ `number` je teda 16-bitové celé číslo bez znamienka (to isté ako `word` v Pascale, `unsigned short` v C/C++).

Čísla sa v rúre samozrejme nemôžu predbiehať, Kvak ich teda bude vyberať v tom istom poradí, v akom ich tam vložil. Rúra je gumená, takže sa do nej čísel zmestí ľubovoľne veľa. Ak nie je povedané ináč, je na začiatku výpočtu rúra prázdna. Okrem rúry má Kvak ešte jeden kotúč žltej pásky, na ktorú môže písať svoj výstup.

Ak sa stane, že pri pokuse vybrať z rúry je rúra prázdna, nastane chyba. Rovnako nastane chyba, ak sa pokúsime deliť nulou, počítať zvyšok po delení nulou, alebo skočiť na neexistujúce miesto. Ak sa program dostane na koniec, Kvak korektne skončí (ako keby na konci programu bola ešte inštrukcia `stop`.)

Pre stručnosť môžeme písať viac príkazov do jedného riadku, v takomto prípade ich od seba treba oddeliť bodkočiarkou.

Príklad 1

Nasledujúci program spočíta a vypíše súčet čísel od 1 do 20.

```
put 20 ; put 0
label start
  get n ; jz n end
  put n ; put n ; put 1 ; add ; sub
  get x ; put x
jump start
label end
print
```

Vždy, keď sa Kvak pri vykonávaní programu dostane k riadku `label start`, budú v rúre práve dve čísla. Keď prvé z nich označíme n, hodnota druhého bude súčet $s = (n + 1) + \cdots + 20$. Následne načítame n do registra `n`. Ak je $n = 0$, máme v rúre hľadaný súčet, môžeme ho vypísať a skončiť. V opačnom prípade chceme spraviť dve veci: Prirátať toto n k doteraz získanému súčtu, a následne zmenšiť n o jedna. Po vykonaní príkazov `put n ; put n ; put 1` máme v rúre postupne čísla: s, n, n, 1. Príkaz `add` sčíta prvé dve, po jeho vykonaní je v rúre: n, 1, $n + s$. Po vykonaní ďalšieho príkazu `sub` máme v rúre hodnoty $n + s$ a $n - 1$. To už je skoro to, čo treba, len sú v opačnom poradí. Preto jednu z nich načítame do `x` a znovu vložíme.

Príklad 2

V rúre je neprázdna postupnosť čísel. Napíšeme program, ktorý spočíta a vypíše jej súčet. (Presnejšie, jeho zvyšok po delení 65 536.)

Dokola budeme opakovať nasledujúci postup: Zistíme, či sú v rúre aspoň dve čísla. Ak áno, tie dve, ktoré sú práve na začiatku, sčítame. Ak už zostalo len jedno, je zjavne súčtom všetkých pôvodných čísel. V programe pre Kvak môžeme túto myšlienku implementovať napríklad nasledovne:

```
label cyklus ; get a ; jempty koniec ; put a ; add ; jump cyklus
label koniec ; put a ; print
```

Na začiatku každej iterácie načítame do registra `a` číslo z rúry. Ak je tá v tomto okamihu prázdna, máme v registri `a` hľadaný súčet, stačí ho už len vypísať. Ak nie, číslo z registra `a` vrátime do rúry. V tomto okamihu sú v rúre aspoň dve čísla, a teda môžeme bez obáv zavolať inštrukciu `add`.

Časová zložitosť tohto riešenia je lineárna od počtu čísel, ktoré boli na začiatku v rúre. Totiž každá iterácia cyklu trvá len konštantne veľa krokov a zmenší nám o jedno počet čísel v rúre.

SLOVENSKÁ KOMISIA OLYMPIÁDY V INFORMATIKE

DVADSIATY PIATY ROČNÍK OLYMPIÁDY V INFORMATIKE

Vydala IUVENTA s finančnou podporou Ministerstva školstva SR
Náklad: 40 výtlačkov
Zodpovedný redaktor: Michal Forišek
Sadzba programom LaTeX