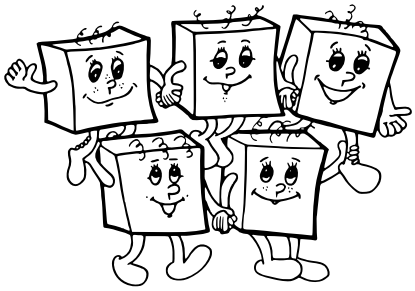
OLYMPIÁDA V INFORMATIKE

NA STREDNÝCH ŠKOLÁCH

http://oi.sk/



tridsiaty ročník

školský rok 2014/2015

zadania krajského kola
kategória A

Priebeh krajského kola

Krajské kolo 30. ročníka Olympiády v informatike, kategória A, sa koná 20. januára 2015 v dopoludňajších hodinách. Na riešenie úloh majú súťažiaci **4 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

Za každú úlohu môžete získať od 0 do 10 bodov.
Pokiaľ nie je v zadaní povedané ináč, najdôležitejšie dve kritériá hodnotenia sú v prvom rade **správnosť** a v druhom rade **efektívnosť** navrhnutého algoritmu. Na výslednom počte bodov sa môže prejaviť 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. Nezávisí pri tom na konštantných faktoroch, len na rádovej rýchlosti rastu tejto funkcie.
V zadaní úloh uvádzame časť „Hodnotenie“, v ktorej nájdete približné limity na veľkosť vstupných údajov. Pod pojmom „efektívne vyriešiť“ chápeme to, že váš program spustený na modernom počítači by mal dať odpoveď nanajvýš do niekoľkých sekúnd.
Údaje z tejto časti zadania by mali slúžiť hlavne na to, aby ste o riešení, ktoré vymyslíte, vedeli približne povedať, koľko bodov zaň dostanete.

A-II-1 Rušenie metra

V Kocúrkove prišiel rok dopravných reforiem. Jej súčasťou je napríklad zbúranie miestneho metra. Metro má stromovú topológiu. Medzi jeho n stanicami teda vedie dokopy $n-1$ tunelov, ktoré sú navrhnuté tak aby bolo metro súvislé – teda aby sa dalo cez tunely prejsť medzi ľubovoľnými dvoma stanicami.
Na zbúranie metra je potrebné postupne jednu po druhej zasypať všetky jeho stanice. Po každom zasypaní stanice však musí platiť, že nezasypaná časť metra je naďalej súvislá (aby sa ešte dala chvíľu používať). Zasypanou stanicou (ani tunelmi, ktoré do nej vedú) už pochopiteľne nemôže metro prechádzať.
Dá sa to vôbec? A ak áno, koľkými rôznymi spôsobmi to vieme spraviť?

Súťažná úloha

Na vstupe je zadaný neorientovaný strom reprezentujúci sieť metra. Vašou úlohou je napísať program, ktorý vypočíta, koľko existuje rôznych poradí rušenia staníc, ktoré spĺňajú vyššie uvedenú podmienku.
Pre veľké stromy by toto číslo mohlo byť veľmi veľké. Pri písaní programu sa tým však nezaťažujte – predpokladajte, že všetky medzivýsledky sa vám zmestia do použitých premenných a že všetky základné aritmetické operácie s nimi viete robiť v konštantnom čase.

Formát vstupu

V prvom riadku vstupu je počet staníc n. Stanice sú očíslované od 1 po n. Nasleduje $n-1$ riadkov. V každom sú dve celé čísla x a y ($1 \leq x, y \leq n$), ktoré hovoria, že stanice x a y sú spojené tunelom.

Formát výstupu

Vypíšte počet poradí búrania staníc, pri ktorých sa nikdy nenaruší súvislosť zvyšných staníc.

Príklad

vstup

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

výstup

```
12
```

Možné poradia búrania sú 2341, 2431, 3241, 3421, 4231, 4321, 2314, 3214, 2413, 4213, 3412, 4312.

Všimnite si, že napríklad nesmieme zbúrať stanicu 1 *ako prvú, lebo by sme po jej zbúraní nevedeli metrom jazdiť medzi ešte nezbúranými stanicami* 2 *a* 3.

Hodnotenie

Plných 10 bodov získate za správne riešenie, ktoré efektívne vyrieši ľubovoľný vstup s $1 \leq n \leq 1\,000\,000$.
Maximálne 8 bodov získate za správne riešenie, ktoré efektívne vyrieši ľubovoľný vstup s $1 \leq n \leq 1\,000$.
Maximálne 4 body získate za správne riešenie, ktoré efektívne vyrieši ľubovoľný vstup s $1 \leq n \leq 10$.

A-II-2 Čert-prezident

V pekle je v posledných dňoch veľmi horúco. Končí sa totižto jedno funkčné obdobie. Vášne vrcholia, davy šalejú, volí sa predsa, kto bude novým prezidentom čertov! Nemyslite si však, že to je jednoduché, správny čert-prezident musí spĺňať prísne kritériá:

- všetci ostatní čerti sa ho boja
- on sa nebojí žiadneho iného čerta

Pomôžte pekelnej volebnej komisii rozhodnúť, či sa práve teraz v pekle nachádza vhodný kandidát na čerta-prezidenta.

Súťažná úloha

V pekle žije n čertov očíslovaných od 1 do n. Keďže vy (zatiaľ?) v pekle nežijete, neviete nič o tom kto sa koho bojí. Aby ste to zistili, budete zadávať otázky typu `boji_sa(a,b)?` pre dvoch rôznych čertov s číslami a a b. Volebná komisia vyšle služobníka, ktorý vám dá odpoveď, buď `pravda`, alebo `nepravda`. O strachu môžete predpokladať len to, že to, kto sa koho bojí, sa nikdy nemení. Inak už nepredpokladajte nič. Je napríklad možné, že sa dvaja čerti boja navzájom, rovnako ako je možné, že sa ani jeden nebojí toho druhého.
Navrhnite postup kladenia jednotlivých otázok tak, aby sa dalo čo najskôr určiť, či sa v pekle nachádza vhodný kandidát na čerta-prezidenta.

Oproti bežným úlohám nás v tejto úlohe bude zaujímať **presný** počet zadaných otázok **v najhoršom prípade**, v závislosti od n. Až v druhom rade bude brané do úvahy aj asymptotické správanie sa počtu otázok, ktorý potrebujete, a v treťom rade potom časová zložitosť vášho riešenia.

Formát implementácie

Implementujte funkciu `prezident`. Táto funkcia má jeden argument n: počet čertov v pekle. Vrátiť má hodnotu 0, pokiaľ vhodný kandidát na čerta-prezidenta neexistuje, resp. číslo vhodného kandidáta na prezidenta, ak taký existuje. (Rozmyslite si, že taký kandidát je vždy nanajvýš jeden.) Počas svojho výpočtu môžete volať funkciu `boji_sa(a,b)`, ktorá vráti 1, pokiaľ sa čert a bojí čerta b a 0 v opačnom prípade. Funkcia `boji_sa` má konštantnú časovú zložitosť.

Hlavičky funkcií v C++:

```
int prezident(int n);
bool boji_sa(int a, int b);
```

Hlavičky funkcií v Pascale:

```
function prezident(n : integer) : integer;
function boji_sa(a, b : integer) : boolean;
```

Príklad

Pokiaľ bude vaša funkcia zavolaná ako `prezident(3)` a otázky s odpoveďami budú vyzerať nasledovne:

- `boji_sa(1, 2)` vráti `true`
- `boji_sa(2, 3)` vráti `false`
- `boji_sa(3, 2)` vráti `true`
- `boji_sa(2, 1)` vráti `false`

je jasné, že vhodný kandidát existuje a je ním čert číslo 2.

Hodnotenie

Riešenia, ktoré sa vždy potrebujú opýtať na všetky dvojice čertov, budú hodnotené najviac 3 bodmi. Riešenia, ktoré sú v niektorých situáciách efektívne ale v najhoršom prípade nie, budú hodnotené najviac 4 bodmi. Riešenia, ktoré majú asymptoticky optimálny počet otázok, môžu získať od 7 do 10 bodov podľa presného počtu položených otázok.

A-II-3 Okružná jazda

Po zrušení metra sa v Kocúrkove zhoršila dopravná situácia a zvýšil sa počet dopravných nehôd. Rozhodli sa preto zaviesť v meste jednosmerky. Pustili sa do toho s takým nadšením, že na každej križovatke zostali nanajvýš dve ulice, ktoré sú obojsmerné.[1] To komplikuje život smetiarom, ktorí by radi prešli každou ulicou práve raz.

Súťažná úloha

Je zadaná sieť ulíc s n križovatkami, očíslovanými od 1 po n, a m ulicami, spájajúcimi dvojice križovatiek. Ulice sa pretínajú iba v popísaných križovatkách, všetky ostatné kríženia sú vyriešené mimoúrovňovo. Každá ulica je označená či je jednosmerná, a ak áno, tak ktorým smerom. Do každej križovatky vedú nanajvýš dve ulice, ktoré nie sú jednosmerné.
Smetiari by chceli začať na križovatke 1, prejsť **každou ulicou práve raz** a skončiť znovu na križovatke 1. Jednosmernou ulicou smú ísť len v povolenom smere. Obojsmernou ulicou chcú ísť len raz (teda nie raz každým smerom). Zistite, či okružná trasa pre smetiarov existuje. Ak áno, jednu takú trasu nájdite.

Format vstupu

Prvý riadok vstupu obsahuje dve prirodzené čísla n a m, oddelené medzerou ($n \geq 1$, $m \geq 0$): počet križovatiek a počet ulíc. Každý z nasledujúcich m riadkov popisuje jednu ulicu. Obsahuje dve rôzne celé čísla x_i a y_i (čísla križovatiek, spojené touto ulicou) a písmeno s_i (typ ulice). Ak $s_i =$ `J`, tak je ulica jednosmerná a je povolené prejsť len v smere z x_i do y_i. Ak $s_i =$ `O`, tak je ulica obojsmerná. Medzi každými dvomi križovatkami vedie nanajvýš jedna ulica.

Format výstupu

Jediný riadok výstupu má obsahovať $m+1$ medzerami oddelených čísel, ktoré udávajú poradie križovatiek také, že začneme a skončíme v križovatke číslo 1, a prejdeme každou ulicou práve raz. Ak je viacero riešení, vypíšte ľubovoľné z nich. Ak žiadne riešenie neexistuje, tak vypíšte text „`neda sa`“.

Príklady

vstup

```
5 6
1 3 J
3 2 J
2 1 J
2 4 O
5 4 O
2 5 O
```

výstup

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

vstup

```
3 3
2 1 J
3 1 J
2 3 O
```

výstup

```
neda sa
```

Hodnotenie

Plných 10 bodov získajú riešenia, ktoré efektívne vyriešia ľubovoľný vstup s $n \leq 100\,000$ a $m \leq 1\,000\,000$. Až 8 bodov získajú riešenia, ktoré efektívne vyriešia ľubovoľný vstup s $n \leq 1\,000$. Čiastočné riešenia, ktoré predpokladajú, že s každou križovatkou susedí nanajvýš jedna obojsmerná ulica, získajú 5 – 7 bodov v závislosti od efektivity. Čiastočné riešenia, ktoré predpokladajú, že všetky ulice sú jednosmerné, získajú nanajvýš 5 bodov.

[1] Križovatky môžu byť ľubovoľne veľké. Pokojne môžeme mať napr. križovatku, z ktorej vedú dve obojsmerné ulice, sedem jednosmerných, a ďalších päť jednosmerných ulíc na ňu vchádza.

A-II-4 Magické siete

K tejto úlohe patrí aj študijný text, ktorý nájdete nižšie. Je identický so študijným textom z domáceho kola.

V zadaniach úloh krajského kola používame nasledovné obmedzenia:

- $\mathrm{ZERO}(x)$ je splnené, ak má premenná x hodnotu 0.
- $\mathrm{NAE}(x, y, z)$ je splnené, ak premenné x, y a z nemajú všetky tri tú istú hodnotu („not all equal“).
- $\mathrm{XOR}_3(x, y, z)$ je splnené, ak má spomedzi premenných x, y a z nepárny počet hodnotu 1.
- $\mathrm{XOR}_4(w, x, y, z)$ je splnené, ak má spomedzi premenných w, x, y a z nepárny počet hodnotu 1.

Súťažná úloha

a) (3 body) Ukážte, že len pomocou obmedzení typu XOR_3 nevieme simulovať XOR_4.
b) (3 body) Ukážte, že pomocou obmedzení typov XOR_3 a ZERO už vieme simulovať XOR_4.
c) (4 body) Ukážte, že len pomocou obmedzení typu NAE vieme simulovať XOR_4.

Študijný text: magické siete

Magická sieť sa skladá z dvoch typov objektov: *premenných* a *obmedzení*.
Premenné budeme zväčša značiť malými písmenami. Každá premenná môže nadobúdať len dve hodnoty: 0 a 1.
Obmedzenia sú vlastne podmienky, ktoré musia premenné spĺňať. Niekoľko príkladov obmedzení:

- Obmedzenie $\mathrm{XOR}_3(x, y, z)$ hovorí, že spomedzi premenných x, y a z musí mať nepárny počet hodnotu 1.
- Obmedzenie $\mathrm{OR}_3(x, y, z)$ hovorí, že spomedzi premenných x, y a z musí mať aspoň jedna hodnotu 1.
- Obmedzenie $\mathrm{EQ}(x, y)$ zase hovorí, že premenné x a y musia mať tú istú hodnotu.

Premenné použité v obmedzení nemusia byť navzájom rôzne. Môžeme napr. použiť obmedzenie $\mathrm{XOR}_3(x, y, x)$.

Niektoré z premenných sú *vstupné*. Sieť sa používa tak, že najskôr nastavíme hodnoty vstupných premenných a potom zošleme magické zaklínadlo. Zaklínadlo sa pokúsi doplniť hodnoty ostatných premenných tak, aby boli súčasne splnené všetky obmedzenia. Ak sa to dá, zaklínadlo jedno také ohodnotenie nájde a magická sieť *prijme zadaný vstup*. V opačnom prípade zaklínadlo zlyhá a sieť *zadaný vstup odmietne*.

Hovoríme, že magická sieť *simuluje obmedzenie O*, ak prijme práve tie ohodnotenia vstupných premenných, pre ktoré je obmedzenie O splnené.

Príklad 1. Uvažujme magickú sieť so 4 premennými a 2 obmedzeniami. Premenné si nazveme a, b, c a d. Obmedzenia v našej sieti budú $\mathrm{XOR}_3(a, b, c)$ a $\mathrm{XOR}_3(b, c, d)$. Premenné a a d nech sú vstupné.

Ľahko nahliadneme, že ak sú obe podmienky súčasne splnené, tak musí platiť $a = d$. Totiž ak poznáme b a c, vieme jednoznačne určiť, akú hodnotu musí mať tretia premenná x aby platilo $\mathrm{XOR}_3(x, b, c)$. No a túto hodnotu musí nadobúdať aj a, aj d.
A naopak: Nech platí $a = d$. Potom vždy vieme nájsť také b a c aby boli obe odmedzenia splnené. Napr. pre $a = d = 0$ zoberieme $b = 0$ a $c = 1$, zatiaľ čo pre $a = d = 1$ zoberieme $b = c = 0$.
Vidíme teda, že naša magická sieť prijme práve tie dvojice (a, d) pre ktoré platí $a = d$. Inými slovami, táto sieť simuluje obmedzenie $\mathrm{EQ}(a, d)$.

Hovoríme, že množina typov obmedzení $\{O_1, O_2, \ldots, O_n\}$ *simuluje* obmedzenie O ak existuje magická sieť ktorá simuluje O a pritom používa len obmedzenia typov O_1 až O_n. Príklad 1 je teda dôkazom toho, že množina typov obmedzení $\{\mathrm{XOR}_3\}$ simuluje obmedzenie typu EQ.

Hovoríme, že obmedzenie $O(x_1, \ldots, x_n)$ je *slabé*, ak existuje len jediná kombinácia hodnôt premenných x_1 až x_n pre ktorú nie je splnené. Slabé obmedzenie budeme zapisovať ako $\mathrm{S}_{h_1 \ldots h_n}$, kde $h_1, \ldots, h_n$ sú hodnoty premenných, pre ktoré nie je splnené.
Teda napr. slabé obmedzenie $\mathrm{S}_{001}(x, y, z)$ je splnené vždy okrem prípadu kedy $x = 0$, $y = 0$ a $z = 1$.
Slabé obmedzenie S_{000} je zhodné s vyššie definovaným obmedzením OR_3.
Množinu všetkých slabých obmedzení s n premennými budeme označovať $\mathcal{S}_n$.

Príklad 2. Dokážeme si, že množina obmedzení $\mathcal{S}_3$ simuluje obmedzenie XOR_3.

Toto je jednoduché. Do našej siete so vstupnými premennými x, y a z dáme štyri obmedzenia: $\mathrm{S}_{000}(x, y, z)$, $\mathrm{S}_{011}(x, y, z)$, $\mathrm{S}_{101}(x, y, z)$ a $\mathrm{S}_{110}(x, y, z)$. Tieto obmedzenia dokopy zakážu všetky kombinácie hodnôt premenných ktoré nechceme prijať.

Príklad 3. Dokážeme si, že množina obmedzení $\mathcal{S}_2$ *nesimuluje* obmedzenie OR_3.

Toto bude o niečo náročnejšie. Aby sa nám ľahšie vyjadrovalo, definujeme si pomocnú funkciu $\mathrm{maj}_3(x, y, z)$. Táto funkcia (nazvaná podľa anglického slova „majority“) vracia na výstup tú z hodnôt 0 a 1, ktorá je na vstupe častejšie. Teda napr. $\mathrm{maj}_3(1, 1, 1) = \mathrm{maj}_3(1, 0, 1) = 1$, ale $\mathrm{maj}_3(0, 0, 1) = 0$.
Samotný dôkaz spravíme sporom. Uvažujme ľubovoľnú sieť, v ktorej používame len obmedzenia z $\mathcal{S}_2$. Premenné v sieti si označíme x_1 až x_n, pričom prvé tri z nich považujeme za vstupné. Predpokladajme, že táto sieť simuluje obmedzenie $\mathrm{OR}_3(x_1, x_2, x_3)$.

Vyskúšajme si do našej siete dosadiť hodnoty $x_1 = 1$, $x_2 = 0$ a $x_3 = 0$. Keďže podmienka $\mathrm{OR}_3(1, 0, 0)$ je splnená, musí aj naša sieť toto ohodnotenie vstupných premenných prijať. Inými slovami, musí existovať nejaké ohodnotenie zvyšných premenných. Tieto hodnoty si nazvime a_1 až a_n.
Podobne musí existovať druhé ohodnotenie premenných $b_1, \ldots, b_n$ ktoré spĺňa všetky obmedzenia v našej sieti a pri ktorom $(b_1, b_2, b_3) = (0, 1, 0)$. No a ešte si nájdeme tretie takéto ohodnotenie $c_1, \ldots, c_n$ pri ktorom $(c_1, c_2, c_3) = (0, 0, 1)$.

Zostrojíme si teraz štvrté ohodnotenie premenných tak, že tieto tri ohodnotenia „spriemerujeme“. Presnejšie, pozrieme sa na ohodnotenie $d_1, \ldots, d_n$ pri ktorom pre každé i platí $d_i = \mathrm{maj}_3(a_i, b_i, c_i)$.

Ukážeme si najskôr, že aj pri tomto novom ohodnotení sú všetky obmedzenia v našej sieti splnené. Každé obmedzenie v našej sieti je tvaru $\mathrm{S}_{h_1 h_2}(x_i, x_j)$ pre nejaké hodnoty h_1, h_2 a nejaké premenné x_i, x_j. Uvažujme ľubovoľné takéto obmedzenie a ukážeme si, že ak ho spĺňajú prvé tri ohodnotenia premenných, musí ho spĺňať aj štvrté.
Pozrime sa na dvojice hodnôt (a_i, a_j), (b_i, b_j) a (c_i, c_j). Sú dve možnosti: buď sú niektoré dve dvojice rovnaké, alebo sú všetky tri navzájom rôzne.
Ak sú niektoré dve rovnaké, bez ujmy na všeobecnosti nech $(a_i, a_j) = (b_i, b_j)$. Potom zjavne platí aj $(d_i, d_j) = (a_i, a_j)$ a teda je táto podmienka splnená aj pri ohodnotení d.
Ak sú všetky tri dvojice navzájom rôzne, tak vlastne vidíme práve raz každú dvojicu okrem tej jednej zakázanej – teda okrem dvojice (h_1, h_2). Napríklad ak $(h_1, h_2) = (1, 0)$, tak v ohodnoteniach a, b a c máme v nejakom poradí dvojice $(0, 0)$, $(0, 1)$ a $(1, 1)$. No a ľahko zistíme, že v takomto prípade je $d_i = 1 - h_1$ a $d_j = 1 - h_2$, a pre takúto dvojicu (d_i, d_j) je naša podmienka opäť splnená.

Tým sme teda dokázali, že aj pri ohodnotení d sú všetky podmienky v našej sieti splnené. Toto je ale hľadaný spor. Prečo? Lebo sa ľahko presvedčíme, že $d_1 = d_2 = d_3 = 0$. A keďže obmedzenie $\mathrm{OR}_3(0, 0, 0)$ nie je splnené, nesmie ani naša sieť prijať vstup $(0, 0, 0)$.

TRIDSIATY ROČNÍK OLYMPIÁDY V INFORMATIKE

Príprava úloh: Michal Forišek, Askar Gafurov, Vladimír Macko, Jaroslav Petrucha
Recenzia: Michal Forišek
Slovenská komisia Olympiády v informatike
Vydal: IUVENTA – Slovenský inštitút mládeže, Bratislava 2015