Informácie a pravidlá

Pre koho je súťaž určená?

Do **kategórie B** sa smú zapojiť len tí žiaci základných a stredných škôl, ktorí ešte ani v tomto, ani v nasledujúcom školskom roku nebudú končiť strednú školu.
Do **kategórie A** sa môžu zapojiť všetci žiaci (základných aj) stredných škôl.

Odovzdávanie riešení domáceho kola

Riešitelia domáceho kola odovzdávajú riešenia sami, v elektronickej podobe, a to priamo na stránke olympiády: `http://oi.sk/`. Odovzdávanie riešení bude spustené niekedy v septembri.

Riešenia kategórie A je potrebné odovzdať najneskôr **15. novembra 2014**.
Riešenia kategórie B je potrebné odovzdať najneskôr **30. novembra 2014**.

Priebeh súťaže

Za každú úlohu domáceho kola sa dá získať od 0 do 10 bodov. Na základe bodov domáceho kola stanoví Slovenská komisia OI (SK OI) pre každú kategóriu bodovú hranicu potrebnú na postup do **krajského kola**. Očakávame, že táto hranica bude približne rovná **tretine maximálneho počtu bodov**.

V krajskom kole riešitelia riešia štyri teoretické úlohy, ktoré môžu tematicky nadväzovať na úlohy domáceho kola. V kategórii B súťaž týmto kolom končí.

V kategórii A je približne najlepších 30 riešiteľov krajského kola (podľa počtu bodov, bez ohľadu na kraj, v ktorom súťažili) pozvaných do **celoštátneho kola**. V celoštátnom kole účastníci prvý deň riešia teoretické a druhý deň praktické úlohy. Najlepší riešitelia sú vyhlásení za víťazov. Približne desať najlepších riešiteľov následne SK OI pozve na týždňové výberové sústredenie. Podľa jeho výsledkov SK OI vyberie družstvá pre Medzinárodnú olympiádu v informatike (IOI) a Stredoeurópsku olympiádu v informatike (CEOI).

Ako majú vyzerať riešenia úloh?

V praktických úlohách je vašou úlohou vytvoriť program, ktorý bude riešiť zadanú úlohu. Program musí byť v prvom rade korektný a funkčný, v druhom rade sa snažte aby bol čo najefektívnejší.

V kategórii B môžete použiť ľubovoľný programovací jazyk.

V kategórii A musíte riešenia praktických úloh písať v jazyku Pascal, C, alebo C++, Odovzdaný program bude automaticky otestovaný na viacerých vopred pripravených testovacích vstupoch. Podľa toho, na koľko z nich dá správnu odpoveď, vám budú pridelené body. Výsledok testovania sa dozviete krátko po odovzdaní. Ak váš program nezíska plný počet bodov, budete ho môcť vylepšiť a odovzdať znova, až do uplynutia termínu na odovzdávanie.

Presný popis, ako majú vyzerať riešenia praktických úloh (napr. realizáciu vstupu a výstupu), nájdete na webstránke, kde ich budete odovzdávať.

Ak nie je v zadaní povedané ináč, riešenia teoretických úloh musia v prvom rade obsahovať **podrobný slovný popis použitého algoritmu, zdôvodnenie jeho správnosti** a diskusiu o efektivite zvoleného riešenia (t. j. posúdenie časových a pamäťových nárokov programu). Na záver riešenia uveďte program. Ak používate v programe netriviálne algoritmy alebo dátové štruktúry (napr. rôzne súčasti STL v C++), súčasťou popisu algoritmu musí byť dostatočný popis ich implementácie.

Usporiadateľ súťaže

Olympiádu v informatike (OI) vyhlasuje *Ministerstvo školstva SR* v spolupráci so *Slovenskou informatickou spoločnosťou* (odborným garantom súťaže) a *Slovenskou komisiou Olympiády v informatike*. Súťaž organizuje *Slovenská komisia OI* a v jednotlivých krajoch ju riadia *krajské komisie OI*. Na jednotlivých školách ju zaisťujú učitelia informatiky. Celoštátne kolo OI, tlač materiálov a ich distribúciu po organizačnej stránke zabezpečuje IUVENTA v tesnej súčinnosti so Slovenskou komisiou OI.

A-I-1 Medzery

Toto je praktická úloha. Pomocou webového rozhrania odovzdajte ***funkčný, odladený program****.*

Zadania súťažných úloh Olympiády v informatike vyzerajú pekne, lebo sú sádzané v programe TEX. Môžete si napríklad všimnúť, že každý odsek textu je pekne zarovnaný na oba okraje. Tento efekt sa dosiahne tak, že sa v každom riadku zvlášť vhodne zvolí veľkosť medzery medzi slovami. Všetky medzislovné medzery v rámci jedného riadku sú vždy rovnaké, ale v rôznych riadkoch často budú rôzne.

Samozrejme, keby sme mali v riadku primálo slov a medzi nimi obrovské medzery, vyzeralo by to zle. A keby sme sa do riadku snažili natlačiť viac slov ako sa tam zmestí, vyzeralo by to ešte horšie. Program robiaci sadzbu textu teda musí nájsť také rozdelenie slov do riadkov textu, ktoré bude vyzerať čo najlepšie.

Pri skutočnej sadzbe je to celé samozrejme o niečo zložitejšie – môžeme napríklad rozdeliť niektoré slová na konci riadku, či prípadne zohľadniť to, že sa nepatrí nechávať na konci riadku predložky. V našej súťažnej úlohe však takéto detaily zanedbáme a budeme sa zaoberať najjednoduchšou možnou verziou.

Súťažná úloha

Strana má šírku 60. Každý znak má šírku 1. Medzislovné medzery môžu mať ľubovoľnú (nie nutne celočíselnú) šírku, avšak vždy to bude aspoň 1. V každom riadku okrem posledného musia byť aspoň dve slová.
Optimálna dĺžka medzery je 1. Túto dĺžku vždy použijeme v poslednom riadku. Pre každý iný riadok spočítame dĺžku medzier v ňom tak, že voľné miesto rovnomerne prerozdelíme. Formálne, pre veľkosť medzier v riadku platí:

$$\text{veľkosť medzery} = \frac{60 - \text{počet písmen v riadku}}{\text{počet slov v riadku} - 1}.$$

Škaredosť medzery je štvorec jej odchýlky od 1. Formálne teda škaredosť riadku vypočítame nasledovne:

$$\text{škaredosť riadku} = (\text{počet medzier v riadku}) \cdot (\text{šírka medzery} - 1)^2.$$

Všimnite si, že škaredosť posledného riadku je vždy 0.
Celková škaredosť riešenia je súčtom škaredosti jednotlivých riadkov.

Formát vstupu a výstupu

Na vstupe je postupnosť slov, oddelených medzerami a koncami riadkov. Každé slovo je tvorené len malými a veľkými písmenami anglickej abecedy.

Na výstupe vypíšte tú istú postupnosť slov, v tom istom poradí. Výstup sa môže od vstupu líšiť len rozdelením slov do riadkov. Každý riadok výstupu musí mať (vrátane medzier) nanajvýš 60 znakov. V každom riadku výstupu okrem posledného musia byť aspoň dve slová. V rámci každého riadku po sebe idúce slová oddeľte jednou medzerou; za posledným slovom v riadku medzeru nevypisujte.

Hodnotenie

Je 10 testovacích vstupov, za každý môžete získať nanajvýš 1 bod. Body môžete získať len za testovacie vstupy, na ktorých vaše riešenie v časovom limite skončí a vypíše korektný výstup.
Počet bodov, ktoré dostanete za konkrétny vstup, vypočítame nasledovným vzorcom:

$$\frac{1 + \text{škaredosť optimálneho riešenia}}{1 + \text{škaredosť tvojho riešenia}}$$

Celkové body za úlohu vypočítame tak, že sčítame tvoje body za všetkých 10 testovacích vstupov a výsledok zaokrúhlime **nadol** na najbližšie celé číslo.

Príklad

vstup

```
Lorem ipsum dolor sit amet consectetur adipiscing elit Morbi sed ullamcorper
laoreet nisl Nunc tincidunt felis sed purus tristique a venenatis leo auctor
Nam vel ultrices enim nec mattis erat Vestibulum vestibulum urna quis
convallis accumsan sapien tortor aliquet massa a mattis ipsum lorem eget
tellus Praesent sollicitudinidorcivenenatis pulvinar Aliquam vitae
ullamcorper diam
```

optimálny výstup

```
Lorem ipsum dolor sit amet consectetur adipiscing elit Morbi
sed ullamcorper laoreet nisl Nunc tincidunt felis sed purus
tristique a venenatis leo auctor Nam vel ultrices enim
nec mattis erat Vestibulum vestibulum urna quis convallis
accumsan sapien tortor aliquet massa a mattis ipsum lorem
eget tellus Praesent sollicitudinidorcivenenatis pulvinar
Aliquam vitae ullamcorper diam
```

V prvom riadku výstupu je 8 medzier, každá z nich má dĺžku 1.000.
V druhom riadku je 8 medzier, každá dĺžky 1.125.
Tretí má 8 medzier dĺžky 1.750.
Vo štvrtom je pre zmenu 7 medzier a tie majú dĺžku 1.429.
V piatom riadku je opäť 8 medzier, tentokrát dĺžky 1.375.
V predposlednom, šiestom riadku sú len 4 medzery, každá dĺžky 1.750.
V poslednom, siedmom riadku sú 3 medzery. Tie budú optimálnej dĺžky 1.

Celková škaredosť je teda $8 \cdot 0.000^2 + 8 \cdot 0.125^2 + 8 \cdot 0.750^2 + 7 \cdot 0.429^2 + 8 \cdot 0.375^2 + 4 \cdot 0.750^2 \approx 9.286$.

iný výstup

```
Lorem ipsum dolor sit amet consectetur adipiscing elit Morbi
sed ullamcorper laoreet nisl Nunc tincidunt felis sed purus
tristique a venenatis leo auctor Nam vel ultrices enim nec
mattis erat Vestibulum vestibulum urna quis convallis
accumsan sapien tortor aliquet massa a mattis ipsum lorem
eget tellus Praesent sollicitudinidorcivenenatis pulvinar
Aliquam vitae ullamcorper diam
```

Toto je iný korektný výstup zodpovedajúci tomu istému vstupu. Jeho celková škaredosť je 12.111. (Riadky majú postupne 8, 8, 9, 6, 8, 4 a 3 medzery. Ich dĺžky sú postupne 1.000, 1.125, 1.222, 2.167, 1.375, 1.750 a 1.000.)

Toto riešenie by dostalo $(1 + 9.286)/(1 + 12.111) \approx 0.785$ bodu. Ak by tvoj program rovnako dobre vyriešil všetkých desať testovacích vstupov, dostal by za celú úlohu 7 bodov.

Veľkosť testovacích vstupov

Každý riadok vstupu má nanajvýš 200 znakov.
Každé slovo na vstupe má nanajvýš 29 znakov (takže sa ľubovoľné dve slová zmestia do riadku výstupu).
Celý vstup má dokopy nanajvýš 100 000 slov.
V testovacích vstupoch 1, 2 a 3 platí, že vstup má nanajvýš 50 slov a optimálny výstup nanajvýš 5 riadkov.
V testovacích vstupoch 4, 5, 6 a 7 platí, že vstup má nanajvýš 1000 slov.

A-I-2 Gaštanové zvieratko

Toto je praktická úloha. Pomocou webového rozhrania odovzdajte ***funkčný, odladený program****.*

Miška si z gaštanov postavila zvieratko. To sa robí tak, že zoberiete n gaštanov a m paličiek (napríklad špajlí alebo zápaliek) a každou paličkou spojíte niektoré dva gaštany. Keď bolo Miškine zvieratko hotové, držalo celé pokope.
Jarko sa rozhodol, že bude Miškine zvieratko postupne rozoberať: každý deň z neho odstráni jeden gaštan (spolu s paličkami ktoré doň viedli). Aby to bolo nenápadnejšie, musí to ale robiť tak, aby sa mu zvieratko nikdy nerozpadlo na viac kusov.

Súťažná úloha

Na vstupe dostanete popis zvieratka. Nájdite jedno možné poradie, v akom odoberať gaštany tak, aby sa vám zvieratko nikdy nerozpadlo na viac kusov. Inými slovami, po každom odobratí gaštanu musí platiť, že všetky gaštany, ktoré sme ešte neodobrali, držia pokope.

Formát vstupu a výstupu

V prvom riadku vstupu sú čísla n a m. Gaštany, ktoré tvoria zvieratko, si očíslujeme od 1 po n. Každý z nasledujúcich m riadkov vstupu popisuje jednu z paličiek: obsahuje dve čísla gaštanov, ktoré táto palička spája. (Každá palička spája dva rôzne gaštany. Každé dva gaštany sú spojené najviac jednou paličkou.)

Na výstup vypíšte jeden riadok a v ňom n medzerami oddelených čísel: čísla gaštanov v poradí, v akom ich má Jarko odoberať. (Vždy existuje aspoň jedno riešenie. Ak existuje správnych riešení viac, vypíšte ľubovoľné jedno z nich.)

Obmedzenia a hodnotenie

Je desať sád testovacích vstupov, za každú sa dá získať jeden bod.
V sadách číslo 1 až 5 platí $n \leq 1\,000$ a $m \leq 10\,000$.
V sadách číslo 6 až 10 platí $n \leq 1\,000\,000$ a $m \leq 10\,000\,000$.

Príklady

vstup

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

výstup

```
1 2 3 4 5
```

Niektoré iné správne riešenia:
5 4 3 2 1
5 1 4 2 3
1 2 5 3 4

vstup

```
7 10
1 4
4 5
5 1
5 2
2 3
3 4
2 6
5 7
2 7
6 7
```

výstup

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

A-I-3 Stratil sa paket

Toto je teoretická úloha. Pomocou webového rozhrania odovzdajte súbor vo formáte PDF, obsahujúci riešenie, spĺňajúce požiadavky uvedené v pravidlách.

Kedysi sme mali na jednom počítači veľký súbor. Chceli sme ho cez sieť preniesť na iný počítač. Rozdelili sme ho pri tom na n paketov, očíslovaných od 0 po $n-1$. Každý z paketov mohol putovať cez sieť inou cestou, a tak na cieľový počítač mohli doraziť v úplne ľubovoľnom poradí. Navyše sa obávame, že sa nám po ceste zopár paketov stratilo.

Súťažná úloha, podúloha A

Dané je číslo n. Na disku máme súbor a v ňom $n-1$ čísel paketov v poradí, v akom dorazili. Čísla v súbore sú z rozsahu od 0 do $n-1$ a sú všetky navzájom rôzne. Súbor môžete čítať ľubovoľne veľakrát, vždy však len sekvenčne. Môžete taktiež vytvárať a používať nové súbory.

Chceme nájsť číslo chýbajúceho paketu a vyžiadať si jeho opätovné poslanie. Dôležité je však, aby toto celé neovplyvnilo výrazne iné úlohy bežiace na počítači, preto je prvoradým kritériom hodnotenia vašich riešení *množstvo použitej pamäte*. Presnejšie, vaše riešenia budeme hodnotiť podľa nasledovných dvoch kritérií:

1. Asymptotická pamäťová zložitosť musí byť čo najmenšia.
2. Pri danej pamäťovej zložitosti je rýchlejšie riešenie lepšie.

Príklad pre podúlohu A

Pre $n = 8$ by súbor s číslami paketov a zodpovedajúci výstup mohli vyzerať nasledovne:

vstup	výstup
`3 1 4 2 0 7 6`	`5`

Súťažná úloha, podúloha B

Riešte tú istú úlohu ako v podúlohe A, len tentokrát chýbajú dva pakety a treba nájsť čísla oboch. Súbor teda obsahuje $n-2$ navzájom rôznych čísel.

Príklad pre podúlohu B

Pre $n = 8$ by súbor s číslami paketov a zodpovedajúci výstup mohli vyzerať nasledovne:

vstup	výstup
`3 1 4 2 0 6`	`5 7`

Hodnotenie

Vzorovému riešeniu by pre $n = 10^9$ stačilo pár kilobajtov pamäte (do čoho rátame úplne všetko, aj dynamicky alokované premenné a zásobník počas rekurzívnych volaní) a menej ako 5 prečítaní súboru. Podobné riešenia dostanú 10 bodov.
Ak si vystačíte s podobným množstvom pamäte, ale budete potrebovať nejaký súbor čítať povedzme tristokrát, môžete počítať so 7 bodmi.
Ak budete riešiť len podúlohu A, môžete získať nanajvýš 5 bodov.

A-I-4 Magické siete

Toto je teoretická úloha. Pomocou webového rozhrania odovzdajte súbor vo formáte PDF, obsahujúci riešenie, spĺňajúce požiadavky uvedené v pravidlách. K tejto úlohe patrí študijný text uvedený na nasledujúcich stranách. Odporúčame najskôr prečítať ten a až potom sa vrátiť k samotným súťažným úlohám. Jednotlivé podúlohy spolu nesúvisia, môžete ich riešiť v ľubovoľnom poradí.

Súťažná úloha, podúloha A

Nech $\mathrm{OR}_2(x, y)$ je obmedzenie, ktoré je splnené, ak aspoň jedna z premenných x a y má hodnotu 1.
Nech $\mathrm{NAE}(x, y, z)$ je obmedzenie, ktoré je splnené, ak premenné x, y a z nemajú všetky tri tú istú hodnotu.
Nech $\mathrm{ONE}(x)$ je obmedzenie, ktoré je splnené, ak má premenná x hodnotu 1.

Nájdite sieť, ktorá simuluje OR_2 pomocou obmedzení typov NAE a ONE.

Súťažná úloha, podúloha B

Nájdite sieť, ktorá simuluje EQ len pomocou obmedzení typu NAE.

Súťažná úloha, podúloha C

Dokážte, že len pomocou obmedzení typu NAE nevieme simulovať OR_3.

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: Vladimír Boža, Michal Forišek
Recenzia: Michal Forišek
Slovenská komisia Olympiády v informatike
Vydal: IUVENTA – Slovenský inštitút mládeže, Bratislava 2014