64. ročník Matematické olympiády, úlohy domácího kola kategorie P

Rozhledy matematicko-fyzikální, Vol. 89 (2014), No. 2, 42–49

Persistent URL: http://dml.cz/dmlcz/146577

64. ročník Matematické olympiády, úlohy domácího kola kategorie P

Úlohy P-I-1 a P-I-2 jsou praktické, vaším úkolem v nich je vytvořit a odladit efektivní program v jazyce Pascal, C nebo C++. Řešení těchto dvou úloh odevzdávejte ve formě zdrojového kódu přes webové rozhraní přístupné na stránce http://mo.mff.cuni.cz/submit/, kde také naleznete další informace. Odevzdaná řešení budou automaticky vyhodnocena pomocí připravených vstupních dat a výsledky vyhodnocení se dozvíte krátce po odevzdání. Pokud váš program nezíská plný počet 10 bodů, můžete své řešení opravit a znovu odevzdat.

Úlohy P-I-3 a P-I-4 jsou teoretické. V úloze P-I-3 je vaším úkolem nalézt efektivní algoritmus řešící zadaný problém. Řešení úlohy se skládá z popisu navrženého algoritmu, zdůvodnění jeho správnosti (funkčnosti) a také odhadu časové a paměťové složitosti. Součástí řešení úlohy P-I-3 je i zápis navrženého algoritmu ve formě zdrojového kódu nebo pseudokódu. V úloze P-I-4 je potřeba popsat požadovanou magickou síť a dokázat, že splňuje zadané vlastnosti; pokud požadovaná síť neexistuje, je nutné její neexistenci zdůvodnit. Řešení obou teoretických úloh odevzdávejte ve formě souboru typu PDF přes výše uvedené webové rozhraní.

Řešení všech úloh můžete odevzdávat do 15. listopadu 2014. Opravená řešení a seznam postupujících do krajského kola najdete na webových stránkách olympiády na adrese *http://mo.mff.cuni.cz/,* kde jsou také k dispozici další informace o kategorii P.

P-I-1 Mezery

Textové editory a obdobné programy pro přípravu textů se snaží, aby jimi vytvořené texty vypadaly esteticky co nejlépe. Podíváte-li se například na toto zadání, všimnete si, že konce všech řádků v odstavci kromě posledního jsou zarovnané přesně pod sebou. Aby toto bylo možné, ne všechny mezery mají stejnou šířku. Je ovšem nežádoucí, aby na některém řádku byly mezery příliš široké nebo naopak příliš úzké. Proto může být vhodné také změnit zalomení řádků. Editory samozřejmě berou do úvahy i další možnosti a omezení (rozdělování slov, řádek by neměl končit předložkou apod.), kterými se ale pro účely této úlohy nebudeme zabývat.

Soutěžní úloha

Je zadán odstavec textu. Určete jeho rozdělení na řádky tak, aby velikosti mezer byly co nejbližší jejich optimální hodnotě.

Formát vstupu

Vstupem je posloupnost slov (složených pouze z malých a velkých písmen latinské abecedy) oddělených mezerami či konci řádků. Každý řádek obsahuje nanejvýš 200 znaků, každé slovo má nejvýše 29 písmen. Celý text má nejvýše 100 000 slov.

Formát výstupu

Výstupem je posloupnost vstupních slov ve stejném pořadí, lišící se pouze rozložením na řádky. Délka každého řádku výstupu smí být nejvýše 60 znaků (včetně mezer) a na každém řádku kromě posledního musí být alespoň dvě slova.

Bodování

Pokud vaše řešení neskončí pro některý z testovacích vstupů v časovém limitu či jeho výstup nesplňuje dané podmínky, nezíská za tento testovací vstup žádné body. Jinak pro každý řádek výstupu kromě posledního určíme šířku mezer tímto způsobem (který předpokládá, že délka řádku je 60 a všechna písmena mají stejnou šířku 1):

$$\text{šířka mezer} = \frac{60 - \text{počet písmen na řádku}}{\text{počet slov na řádku} - 1}.$$

Dále určíme chybu řádku tímto způsobem (za každou mezeru započítáme druhou mocninu její odchylky od optimální šířky 1; díky použití druhé mocniny podstatně více penalizujeme neesteticky vypadající velké mezery): $\text{chyba} = (\text{počet slov na řádku} - 1) \cdot (\text{šířka mezer} - 1)^2$. Chyba celého odstavce je součet chyb jeho řádků. Vaše řešení pak za tento testovací vstup obdrží

$$\frac{1 + \text{chyba optimálního řešení}}{1 + \text{chyba vašeho řešení}} \quad \text{bodů.}$$

Body za všech 10 testovacích vstupů pak sečteme a výsledek zaokrouhlíme na nejbližší celé číslo.

V sedmi z testovacích vstupů bude mít text nejvýše 1 000 slov. Ve třech z nich bude mít optimální řešení nejvýše 5 řádků.

Příklad

Pro vstup

```
Lorem ipsum dolor sit amet consectetur adipiscing elit Morbi sed ullamcorper
mi quis pretium diam Praesent volutpat at dolor nec mollis Praesent at odio
non metus ornare egestas Nullam justo libero luctus vitae tortor vel posuere
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
```

je optimální následující 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
```

Jeho první řádek má 8 mezer délky 1.000, druhý má 8 mezer délky 1.125, třetí má 8 mezer délky 1.750, čtvrtý má 7 mezer délky 1.429, pátý má 8 mezer délky 1.375 a šestý (tedy předposlední) má 4 mezery délky 1.750. Celková chyba tedy je

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

Naproti tomu vydá-li vaše řešení 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
```

s celkovou chybou 12.111 (první řádek 8 mezer délky 1.000, druhý 8 mezer délky 1.125, třetí 9 mezer délky 1.222, čtvrtý 6 mezer délky 2.167, pátý 8 mezer délky 1.375 a šestý 4 mezery délky 1.750), získá za něj 0.785 bodu. Kdyby stejně úspěšně řešilo všech 10 testovacích vstupů, získá 8 bodů.

P-I-2 Rušení stanic

Radní v Kocourkově řeší problém, proč jsou poslední dobou obyvatelé města tak bledí. Napadlo je, že za to může nedostatek sluníčka. Proto se rozhodli, že zruší ve městě metro a lidé budou nuceni strávit víc času venku. Aby však změna nebyla pro obyvatele příliš náhlá, rozhodli se, že budou metro rušit postupně po jednotlivých stanicích. Navíc by chtěli, aby zbylá síť metra i po zrušení několika stanic zůstala souvislá – tedy aby se mezi každými dvěma zbylými stanicemi bylo možné dopravit (třeba i s několika přestupy), aniž by bylo potřeba projet zrušenou stanicí. Pomůžete radním rozhodnout, v jakém pořadí mají rušit stanice metra?

Soutěžní úloha

Je zadána souvislá síť metra s N stanicemi očíslovanými od 1 do N a s M obousměrnými linkami mezi některými dvojicemi stanic. Vypište čísla všech stanic v pořadí, ve kterém by bylo možné je ze sítě postupně odebírat, aniž by zbývající síť v kterémkoli okamžiku přestala být souvislá.

Formát vstupu

Program čte vstupní data ze standardního vstupu. První řádek obsahuje dvě celá čísla N, M oddělená mezerou ($N \geq 1$, $M \geq 0$), udávající po řadě počet stanic a počet linek v síti. Každý z následujících M řádků obsahuje dvě celá čísla od 1 do N, udávající dvojici stanic, které jsou spojeny linkou. Můžete předpokládat, že žádná linka se ve vstupu neobjeví dvakrát a žádná stanice není spojena sama se sebou.

Formát výstupu

Program vypíše na standardní výstup jediný řádek. Na něm bude N mezerami oddělených čísel udávajících pořadí, v jakém mají být stanice odstraněny. Pokud je více možných řešení, vypište libovolné z nich.

Příklady

Vstup:	*Výstup:*

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

Další možná řešení jsou např. `5 4 3 2 1`, `5 1 4 2 3` *nebo* `1 2 5 3 4`.

Vstup:	*Výstup:*
7 11 1 4 4 5 5 1 5 2 2 3 3 4 2 6 5 7 2 7 6 7	1 4 5 3 7 2 6

Uvedený výstup je jednou z mnoha správných odpovědí.

Bodování

Plných 10 bodů obdrží správné řešení, které efektivně vyřeší libovolný vstup s $N \leq 1\,000\,000$ a $M \leq 10\,000\,000$. Až 5 bodů získá správné řešení, které efektivně vyřeší libovolný vstup s $N \leq 1\,000$ a $M \leq 10\,000$.

P-I-3 Ztracený paket

Při přenosu velkých souborů po síti je nutné je rozdělit na menší kusy (pakety), které jsou doručovány samostatně. Při doručování není zaručeno zachování pořadí paketů. Aby je přijímající počítač dokázal správně spojit, pakety jsou očíslovány od 1 do n dle pořadí, v jakém po sobě následují v souboru.

Na přijímající počítač zatím dorazilo $n - 1$ z n odeslaných paketů, jeden se tedy ztratil. Abychom mohli odesílající počítač požádat o jeho znovuzaslání, potřebujeme určit číslo tohoto ztraceného paketu. Přijaté pakety jsme zatím neměli čas nijak zpracovat, máme je pouze uloženy na disku v pořadí, v jakém dorazily. Paketů je mnoho, nemůžeme si proto všechna jejich čísla uložit do paměti. Čtení dat z disku je ovšem pomalé a chceme jej minimalizovat.

Formát vstupu

Váš program čte data ze vstupního souboru `pakety.txt`. Na jeho prvním řádku je přirozené číslo n ($1 \leq n \leq 1\,000\,000\,000$). Na každém z $n-1$ následujících řádků je jedno přirozené číslo v rozsahu 1 až n včetně, udávající číslo paketu. Každé číslo paketu se v souboru vyskytuje nejvýše jednou, pořadí čísel paketů ve vstupním souboru může být libovolné.

Formát výstupu

Výstupem je přirozené číslo v rozsahu 1 až n včetně, které se nevyskytuje mezi čísly paketů ve vstupním souboru.

Příklad

Vstup:	*Výstup:*
6 2 4 1 6 5	3

Bodování

Přijatelná jsou pouze řešení, jejichž proměnné na běžném počítači používají nanejvýš 10 kB paměti (do tohoto limitu se počítá i dynamicky alokovaná paměť a zásobník návratových hodnot v rekurzivních programech; naopak se do něj nepočítají vnitřní proměnné standardních knihoven apod.).

Plných 10 bodů obdrží takové řešení, které přečte vstupní soubor právě jednou a nepoužívá žádné další pomocné soubory. Až 7 bodů může obdržet řešení, které přečte vstupní soubor (či pomocné soubory srovnatelné velikosti) nejvýše 300krát. Libovolné jiné přijatelné řešení může obdržet až 4 body.

P-I-4 Magická síť

K této úloze se vztahuje studijní text uvedený na následujících stranách. Doporučujeme nejprve prostudovat studijní text a až potom se vrátit k samotným soutěžním úlohám.

Soutěžní úloha

Úkol 1: Nechť NAE(x, y, z) je omezení předepisující, že proměnné x, y a z nemají všechny stejnou hodnotu. Nechť ONE(x) je omezení předepisující, že proměnná x má hodnotu 1. Nalezněte síť, používající pouze omezení typu NAE a ONE, která simuluje OR.

Úkol 2: Nalezněte síť, používající pouze omezení typu NAE, která simuluje EQ(a, b), kde a a b jsou navzájem různé proměnné.

Úkol 3: Ukažte, že pomocí NAE nelze simulovat OR.

Studijní text

Magická síť se skládá z *omezení* a *proměnných*. Každá proměnná může nabývat hodnot 0 nebo 1. Omezení pak předepisují podmínky, které ohodnocení proměnných musí splňovat. Např. omezení $\textsc{xor}(x, y, z)$ předepisuje, že z proměnných x, y a z jich lichý počet musí mít hodnotu 1, omezení $\textsc{or}(x, y, z)$ předepisuje, že alespoň jedna z x, y a z musí mít hodnotu 1, omezení $\textsc{eq}(x, y)$ předepisuje, že x a y musí mít stejnou hodnotu, a podobně. Proměnné se v jednom omezení mohou opakovat.

Některé z proměnných jsou *vstupní* a můžeme jim nastavit konkrétní hodnotu. Po seslání příslušného zaklínadla se pak ostatním proměnným nastaví takové hodnoty, aby všechna omezení byla splněna. Pokud žádná taková volba hodnot neexistuje, zaklínadlo nás na to upozorní. V prvním případě říkáme, že magická síť zadaný vstup *přijímá*, ve druhém ho *odmítá*. Magická síť *simuluje omezení* O, jestliže přijímá právě stejné hodnoty proměnných jako omezení O.

Příklad 1: Magickou síť zapisujme jako seznam typů omezení, k nimž do závorek budeme připisovat proměnné, na které jsou aplikovány. Síť $\textsc{xor}(a, b, c)$, $\textsc{xor}(b, c, d)$ tedy vynucuje, že lichý počet z proměnných a, b a c má hodnotu 1 a že lichý počet z proměnných b, c a d má hodnotu 1.

Nechť a a d jsou vstupní proměnné této sítě. Jestliže a má hodnotu 0, pak právě jedna z proměnných b a c musí mít hodnotu 1, a proto d musí mít hodnotu 0. Naopak, má-li a hodnotu 1, pak hodnota proměnné b musí být stejná jako hodnota proměnné c, a proto d musí mít hodnotu 1.

Tato magická síť tedy přijímá právě ty vstupy, kde a a d mají stejnou hodnotu, a simuluje tedy omezení $\textsc{eq}(a, d)$.

Obecněji: Typicky nás bude zajímat, která omezení jdou vyjádřit pomocí jiných. Říkáme, že množina typů omezení $\{O_1, O_2, \ldots, O_n\}$ *simuluje* omezení O, jestliže existuje magická síť používající pouze omezení typu O_1, O_2, …, O_n, která simuluje O. Příklad 1 tedy ukazuje, že XOR simuluje EQ.

Omezení $O(x_1, \ldots, x_n)$ nazveme *slabé*, jestliže zakazuje právě jednu kombinaci hodnot proměnných $x_1, \ldots, x_n$. Slabé omezení budeme zapisovat jako $S_{h_1 h_2 \ldots h_n}$, kde h_1, …, h_n jsou hodnoty proměnných, které zakazuje. Třeba omezení $S_{001}(x, y, z)$ je splněno, jestliže $x = 1$ nebo $y = 1$ nebo $z = 0$, a omezení S_{000} je stejné jako OR. Nechť $\textsc{sat}_n$ označuje množinu všech slabých omezení s právě n proměnnými.

Příklad 2: SAT_3 simuluje XOR, jelikož síť

$$S_{000}(x,y,z),\ S_{011}(x,y,z),\ S_{101}(x,y,z),\ S_{110}(x,y,z)$$

zakazuje všechny kombinace hodnot proměnných x, y a z, v nichž se hodnota 1 vyskytuje suděkrát.

Příklad 3: Množina omezení SAT_2 nesimuluje OR. Abychom to dokázali, zaveďme si nejprve funkci maj se třemi vstupy. Ta bude vracet ten vstup, který se vyskytuje nejčastěji (proto se jí také někdy říká *majorita*). Tedy třeba $\text{maj}(1,1,1) = \text{maj}(1,0,1) = 1$ a $\text{maj}(0,0,1) = 0$.

Uvažujme nyní libovolnou síť s omezeními z množiny SAT_2. Nechť $x_1, \ldots, x_n$ jsou proměnné této sítě. Řekněme, že by tato síť simulovala $\text{OR}(x_1, x_2, x_3)$. Jelikož omezení OR je splněno pro hodnoty 1, 0, 0, existuje nějaké přiřazení hodnot proměnným, které splňuje všechna omezení, x_1 má hodnotu 1 a x_2 a x_3 mají hodnoty 0. Nechť a_i označuje hodnotu proměnné x_i v tomto přiřazení, pro $i = 1, \ldots, n$. Obdobně existuje přiřazení s hodnotami b_i splňující všechna omezení takové, že $b_2 = 1$ a $b_1 = b_3 = 0$, a přiřazení s hodnotami c_i splňující všechna omezení takové, že $c_3 = 1$ a $c_1 = c_2 = 0$.

Uvažme ohodnocení s hodnotami $d_i = \text{maj}(a_i, b_i, c_i)$. Tvrdíme, že d_i také splňuje všechna omezení: Mějme nějaké omezení $S_{h_1 h_2}(x_i, x_j)$ ze sítě. Pokud $a_i = b_i$ a $a_j = b_j$, pak $d_i = \text{maj}(a_i, a_i, c_i) = a_i$ a $d_j = \text{maj}(a_j, a_j, c_j) = a_j$ splňuje podmínku $S_{h_1 h_2}$, protože ji splňuje a_i a a_j. Proto předpokládejme, že $a_i \neq b_i$ nebo $a_j \neq b_j$, a obdobně $a_i \neq c_i$ nebo $a_j \neq c_j$ a stejně tak $b_i \neq c_i$ nebo $b_j \neq c_j$. Ze symetrie mezi i a j a mezi ohodnoceními a, b, c stačí uvažovat případ, že $a_i \neq b_i$ a $a_i \neq c_i$. Z toho odvodíme, že $b_i = c_i$, a proto $b_j \neq c_j$. Díky symetrii mezi b a c pak stačí uvažovat případ, že $a_j = b_j$ a $a_j \neq c_j$. Pak ale $\text{maj}(a_i, b_i, c_i) = \text{maj}(a_i, b_i, b_i) = b_i$ a $\text{maj}(a_j, b_j, c_j) = \text{maj}(b_j, b_j, c_j) = b_j$, a proto $d_i = b_i$ a $d_j = b_j$ splňuje podmínku $S_{h_1 h_2}$.

Povšimněme si, že $d_1 = \text{maj}(1,0,0) = 0$ a obdobně $d_2 = d_3 = 0$. Ale omezení $\text{OR}(x_1, x_2, x_3)$ předepisuje, že alespoň jedna z proměnných x_1, x_2 a x_3 má hodnotu 1, a proto v každé magické síti simulující $\text{OR}(x_1, x_2, x_3)$ musí přiřazení hodnot d_i proměnným porušovat nějaké omezení. Uvažovaná síť tedy nesimuluje $\text{OR}(x_1, x_2, x_3)$.