Korešpondenčný seminár z programovania
XX. ročník, 2002/2003
Katedra vyučovania informatiky FMFI UK,
Mlynská Dolina, 842 48 Bratislava

KSP finančne podporuje MICROSTEP-MIS spol. s r.o.

Príklady 2. kola zimnej časti

Milé deti,
Práve sa vám dostali do rúk zadania druhého kola KSP. Podľa poradia po tomto kole budeme pozývať účastníkov na jarné sústredenie. Preto zapojte všetky svoje mozgové bunky a s chuťou sa vrhnite do riešenia nových piatich úloh, ktoré sme pre vás pripravili. Svoje riešenia nám posielajte do **6. januára 2003** a nezabudnite priložiť známky v hodnote 15 Sk,-.

Myslím, teda spím.
KSPáci

1. Opravovanie

15 bodov

Janka sa rozhodla vyskúšať si prácu učiteľky. Práca to bola veľmi milá, ale iba po prvú písomku. Potom zistila, že tú treba nielen vymyslieť, ale aj opraviť. A to zaberá vééľmi veľa času. Po niekoľkých písomkách sa rozhodla, že to už tak ďalej nejde a vymyslela si fintu. Prečo by mala opravovať ona, veď to môžu urobiť za ňu rovno žiaci. Žiaci si medzi sebou písomky povymieňajú, Janka napíše na tabuľu správne výsledky a oni si ich poopravujú. Ona potom len rýchlo skontroluje, či nepodvádzali a nechceli si pridať body.

Janka žiakov na písomkách náhodne rozďeľuje do niekoľkých skupín. Kedže je Janka správna pedagogička, rozhodla sa, že každý žiak by mal kontrolovať inú skupinu ako tú, ktorú riešil. Vraj aby mali možnosť vidieť čo najviac rôznych príkladov. Teraz ale stojí celá nešťastná pred žiakmi, ktorý jej tvrdia, že to vôbec nejde. Pomôžte jej čo najrýchlejšie vyriešiť tento problém.

Úloha: Rozdelenie žiakov do skupín môžeme reprezentovať reťazcom znakov 'a' až 'z', pričom znak na i-tej pozícii určuje do ktorej skupiny patrí i-ty žiak. Napíšte program, ktorý pre takýto reťazec zistí, či existuje také preusporiadanie písomiek, aby nikto neopravoval svoju skupinu, teda aby na žiadnej pozícii nezostalo rovnaké písmeko. Ak takéto preusporiadanie existuje, vypíšte ľubovoľné jedno spomedzi nich.

Príklad:

Vstup:	**Výstup:**
`abcabcabc`	`Dá sa: cabcabcab.`
`abababa`	`Nedá sa.`

2. O metre II

13 bodov

Smutní boli pracovníci firmy Tunelár a syn, keď im prišli vaše riešenia. Zistili totiž, že trať sa nedá postaviť tak, aby stanice boli v požadovanom poradí. Preto pán riaditeľ Tunelár rozhodol, že (samozrejme na náklady mesta) dajú vytlačiť nové brožúrky pre vodičov, v ktorých budú stanice v inom poradí.[1] Boja sa však, aby sa im opäť neprihodilo to isté. Preto potrebujú program, ktorý im zistí všetky možné poradia staníc, pre ktoré sa dá postaviť trať metra.

[1] Tlač brožúriek zabezpečuje firma Brat Tunelár so ženou.

Pripomeňme si, ako taká stavba metra vyzerá. Výstavba začala tým, že v meste postavili N staníc metra na rôznych významných miestach. Až potom si hlavný projektant uvedomil, že trasa metra nesmie mať žiadne zákruty (inak by sa nedala dosiahnuť taká neuveriteľná rýchlosť), a teda na pláne bude zaznačená ako veľmi dlhá úsečka, ba priam až priamka. Všetky postavené zastávky však ani omylom neležali na jednej priamke. Hlavný projektant však našiel riešenie – keď už bude metro postavené, každá stanica sa k trase metra pripojí pohyblivými schodami, vedúcimi zo stanice ku metru (kolmo na jeho dráhu).

Úloha: Váš program by mal načítať počet N staníc, ich súradnice x_i, y_i a vypíše všetky možné poradia staníc na trase metra. Stanice sú očíslované od 1 do N v poradí, v akom sú na vstupe. Trasu metra, na ktorej by dve stanice mali ležať v tom istom bode, považujeme za neprípustnú.

Príklad:

Vstup:
$N = 4$
`(0,0), (1,0), (2,0), (3,1)`

Výstup:
Prípustné poradia sú:
(1,2,3,4), (1,2,4,3), (1,4,2,3), (4,1,2,3), (4,3,2,1), (3,4,2,1), (3,2,4,1), (3,2,1,4).

3. O cyklistike na Záhorí

15 bodov

Tohtoročná cyklistická sezóna sa už pre Dávidka skončila.[2] Bicykel už síce odstavil, ale zato v teple svojej izby plánuje cyklotrasy na budúcu sezónu. Rád by na jar podnikol niekoľko výletov do Malaciek, lebo tam majú dobrú a lacnú kofolu. Dávidko je, ako každý, lenivý, preto si vždy vyberá zásadne najkratšie trasy. Vašou úlohou je zistiť, koľko rôznych najkratších trás existuje medzi Bratislavou a Malackami.

Úloha: Daná je cestná sieť medzi mestami a dedinkami Záhoria a okolia. Na vstupe je dané N – počet obcí a M – počet ciest vedúcich medzi nimi. Obce sú očíslované číslami 1 až N, pričom Bratislava má číslo 1 a Malacky majú číslo N. Na vstupe ďalej nasleduje M trojíc čísel x_i, y_i, d_i ($1 \leq i \leq M$, $d_i > 0$) znamenajúcich, že medzi obcami číslo x_i a y_i vedie (obojsmerná) cesta dĺžky d_i kilometrov. Vašou úlohou je zistiť, koľko rôznych najkratších trás vedie z Bratislavy do Malaciek.

Príklad:

Vstup:
$N = 5$, $M = 7$
```
1 2 3
1 3 7
4 5 2
3 5 3
4 2 5
3 4 1
2 3 5
```

Výstup:
Existujú 3 najkratšie trasa medzi Bratislavou a Malackami.

(Prvá trasa vedie cez obce 1, 2, 4, 5; druhá 1, 3, 5; a tretia 1, 3, 4, 5. Každá z nich má dĺžku 10 km.)

4. O Miškovi na ľade

18 bodov

Prišla zima, začalo mrznúť a Miško sa rozhodol, že sa naučí korčuľovať. Po dlhom čase sa mu podarilo pozháňať dostatočne veľké korčule, napchal si do nohavíc vankúš a hurá na jazero. Skúša Miško korčuľovať, skúša, vlastne ani nepadá, len tie korčule sa voľajako nekĺžu. Bodaj by sa kĺzali, keď sa Miško len snehom brodí. Všimol si, že ľudia na niektorých miestach odhrnuli sneh, a tak by sa šiel učiť tam. Ktoré miesto si ale vybrať? Čím väčšie, tým lepšie,

[2] Lebo je zima, až za prsty zachádza.

povedal si Miško a začal hľadať najväčšiu odhrnutú obdĺžnikovú plochu. Aby nám pri tom hľadaní nezamrzol, bolo by mu treba pomôcť.

Úloha: Jazero má tvar obdĺžnika so stranami dĺžky M a N. Na prvom riadku vstupu dostanete čísla M a N. Každý z nasledujúcich M riadkov bude obsahovať N núl alebo jednotiek. Nula značí odhrnutý ľad, jednotka zasnežený ľad. Nájdite obdĺžnik, ktorý je tvorený iba nulami a má zo všetkých takých obdĺžnikov najväčší obsah. Ak je takých obdĺžnikov viac, nájdite ľubovoľný z nich. Vypíšte súradnice ľavého horného a pravého dolného rohu takéhoto obdĺžnika.

Poznámka: Plný počet bodov môže získať iba riešenie, ktorého časová zložitosť bude lineárna od plochy ľadu, teda $O(MN)$. Aj keď sa to možno nezdá, takéto riešenie existuje :-)

Príklad:

Vstup:
```
6 4
0010
1000
0100
0001
0000
0100
```

Výstup:
Ľavý horný roh: 4. riadok, 1. stĺpec
Pravý dolný roh: 5. riadok, 3. stĺpec

5. O celulárnych automatoch II

15 bodov

V druhej časti nášho malého seriálu sa pozrieme na automaty, ktoré už budú môcť mať aj viac ako 2 stavy. Naša sústava automatov bude tentokrát tvorená N rovnakými automatmi, uloženými v jednom rade. Každý automat má teda 2 susedov, až na krajné dva, ktoré majú len jedného.

Pripomeňme si, ako vyzerá výpočet takejto sústavy. Každý automat sa nachádza v niektorom z konečne veľa stavov. Každú sekundu sa každý automat pozrie na svoj stav a na stavy svojich dvoch susedov a podľa tejto informácie sa rozhodne, ako zmení stav. Zavedieme špeciálny stav \$, ktorý vidia krajné automaty ako stav ich chýbajúceho suseda. (Dohodnime sa, že automat nemôže nadobudnúť tento stav a teda sa tváriť, že tam nie je.)

Čo bude takáto sústava automatov vedieť rátať? Na prvý pohľad by sa mohlo zdať, že nie veľa. Každý automat predsa vie len o svojom okolí a keďže má len konečne veľa stavov (t.j. konečnú pamäť), žiaden z nich nie je schopný zapamätať si toho viac. Napriek tomu už takáto jednoduchá sústava toho bude vedieť prekvapivo veľa. (Dokonca všetko to, čo váš počítač vie porátať s lineárnou pamäťou od veľkosti vstupu.)

Nás budú zaujímať operácie s číslami. Naša sústava automatov si vie jednoducho zapamätať prirodzené číslo: Zapíšme ho v dvojkovej sústave ako $a_n ... a_1 a_0$. Automaty budú mať (okrem iného) stavy 0 a 1, pritom i-ty automat sprava bude v stave a_i. S týmto číslom bude naša sústava vedieť robiť takmer ľubovoľné operácie. Akú operáciu robí bude samozrejme závisieť (len) od prechodovej funkcie, čiže od programu, ktorý má každý z automatov.

Príklad. Chceme, aby naša sústava automatov vedela pamätané číslo zväčšiť o 1. Bude fungovať nasledovne: V niektorej sekunde keď sa najpravejší automat pozrie na svojho pravého suseda, namiesto \$ mu povieme $*$. To bude špeciálny stav, znamenajúci, že chceme pamätané číslo zväčšiť o 1. Po niekoľkých krokoch by sa naša sústava mala „ustáliť“ a pamätať si o jedno väčšie číslo. Ako to dosiahnuť? Samozrejme voľbou vhodnej prechodovej funkcie.

(Kvôli šetreniu miestom a zároveň lepšej prehľadnosti budeme časť prechodovej funkcie: „Ak môj ľavý sused je v stave U, ja som v stave V a môj pravý sused je v stave W, zmením stav na X“ zapisovať zjednodušene $UVW \to X$. Stavy automatov v konkrétnej sekunde budeme zapisovať ako jeden reťazec, t.j. napr. zápis \$$A011AB$\$ znamená, že naša sústava má 6 automatov, najľavejší automat je v stave A, jeho sused je v stave 0, atď. Znaky \$

predstavujú stavy, ktoré vidia krajné automaty. Tento zápis zdôrazňuje, že na prechodovú funkciu sa môžeme dívať ako na sadu prepisovacích pravidiel – každú sekundu si pre každý automat nájdeme to pravidlo prechodovej funkcie, ktoré „naň pasuje“ a potom podľa týchto pravidiel všetkým automatom naraz zmeníme stavy.)

Náš automat bude môcť nadobudnúť stavy 0, 1, A a B. Načo budú slúžiť stavy A a B? Prirátanie 1 k číslu v dvojkovom zápise vyzerá tak, že posledná 0 sa zmení na 1 a všetky 1 napravo od nej sa zmenia na 0. Toto bude postupne robiť aj naša sústava. Postupne idúc sprava budú automaty meniť svoj stav z 1 na A, až kým nenájdeme prvý v stave 0. Ten zmení svoj stav na 1, zároveň jeho pravý sused zmení stav z A na B. Teraz postupne všetky automaty v stave A idúc zľava doprava zmenia stav na 0 a skončili sme.

Jeho prechodová funkcia bude vyzerať nasledovne:

$xyz \to y$ pre $x, z \in \{0, 1, \$\}$, $y \in \{0, 1\}$ (ak sú v okolí samé čísla, nič nerobíme)

$x0* \to 1$ pre $x \in \{0, 1, \$\}$ (ak je posledná cifra 0, len ju zmeníme na 1)

$x1* \to A$, $x1A \to A$, $yAz \to A$ pre $x \in \{0, 1, \$\}$, $y \in \{1, A\}$, $z \in \{A, \$\}$ (postupne sprava prefarbujeme 1 na A, ak už sme v stave A, ostávame v ňom)

$x0A \to 1$, $0Ay \to B$ pre $x \in \{0, 1, A, \$\}$, $y \in \{A, \$\}$ (prvú 0 zmeníme na 1 a v tej istej sekunde sused prejde do stavu B)

$BAy \to B$, $xBy \to 0$ pre $x \in \{0, 1\}$, $y \in \{A, \$\}$ (pravý sused B sa zmení na B, B na 0)

Ukážeme si výpočet našej sústavy na vstupe 100111:

$\$100111\$ \Rightarrow \$100111* \Rightarrow \$10011A\$ \Rightarrow \$1001AA\$ \Rightarrow \$100AAA\$ \Rightarrow \$101BAA\$ \Rightarrow \$1010BA\$ \Rightarrow \$10100B\$ \Rightarrow \$101000\$$ a v tomto stave už sústava zostane.

Úloha: Určte (konečnú!) množinu stavov automatu a napíšte jeho prechodovú funkciu tak, aby po zadaní príkazu (opäť zmenou „pravej zarážky“ z \$ na $*$) sústava vynásobila zapamätané číslo tromi. Môžete predpokladať, že nenastane pretečenie, t.j. výsledok si je sústava schopná pamätať. Napríklad keď sústavu spustíme v stave \$00001011\$, mala by sa po niekoľkých sekundách ustáliť v stave \$00100001\$.

Nezabudnite, že pri písaní prechodovej funkcie neviete, koľko vašich automatov vedľa seba naskladáme – tento počet je určený dĺžkou vstupného čísla. Sústava zložená z ľubovoľného počtu vašich automatov by mala fungovať podľa zadania.

Pri návrhu prechodovej funkcie vám môže (a nemusí) pomôcť program, simulujúci sústavu vašich automatov. Ak si aj takýto program napíšete, neposielajte nám ho. Ako riešenie stačí **poriadne popísaná** množina stavov a prechodová funkcia vášho automatu. Ak to pomôže prehľadnosti vášho riešenia, stavy nemusíte značiť iba písmenami, napr. L'_{47} je skvelý názov pre stav.