Korešpondenčný seminár z programovania
XXIII. ročník, 2005/06
Katedra základov a vyučovania informatiky FMFI UK,
Mlynská Dolina, 842 48 Bratislava

KSP finančne podporujú: aSc – Applied Software Consultants spol. s r.o.
Gnome spol. s r.o.
MICROSTEP-MIS spol. s r.o.

Príklady 1. kola letnej časti

Milé naše riešiteľky, mílí naši riešitelia!

Leto je ešte ďaleko, no my sa tým nedáme zastrašiť a prinášame vám už teraz zadania 1. kola letnej časti 23. ročníka KSP. V prípade, že riešite KSP po prvý raz, odporúčame vám začať s kategóriou KSP-Z, ktorej zadania nájdete o pár strán ďalej. A vy ostrieľaní veteráni sa už môžete pustiť do čítania a riešenia. (Hlavne piatu úlohu sa neoplatí nechávať na poslednú chvíľu!)

Svoje riešenia nám posielajte najneskôr do **13. marca 2006**, tí šťastnejší z vás nám ich môžu odovzdať aj pri príchode na sústredenie. K riešeniam nezabudnite priložiť nejaké tie známky, tak rádovo v hodnote 15 Sk.

Dovtedy sa chodí s džbánom po vodu, kým niekto nevymyslí vodovod.

KSPáci

1. O lúpeži

15 bodov

Zlatokopi Santo a Banto žijú vo Vyšnej Klondike. Kedysi tu bolo veľa zlata, ale teraz je už skoro všetko vyťažené. Prednedávnom prišli o prácu a pomaly sa im míňajú peniaze. Lovom rýb sa im živiť nechce, preto sa rozhodli vykradnúť banku.

Podarilo sa im totiž zistiť, že ak ukradnú sumu neprevyšujúcu M korún, banka si nič nevšimne a nebudú mať z toho žiadne problémy. Práve teraz sú v trezore a chcú si odtiaľ zobrať čo najviac peňazí. Preto by ste im mali pomôcť.

Úloha: Na vstupe sú čísla N, M – počet druhov bankoviek a maximálna suma, ktorú si môžu odniesť. Ďalej nasleduje N dvojíc celých čísel h_i a p_i, kde h_i je hodnota i-teho typu bankoviek a p_i je počet takýchto bankoviek v trezore. Vašou úlohou je zistiť, aká najvyššia suma menšia ako M sa dá z trezoru ukradnúť.

Rádové veľkosti čísel zo vstupu: N je do 10, M je rádovo 100 000, žiadne z čísel h_i a p_i neprekročí 1000.

Vstup:	**Výstup:**
$N = 3, M = 47$ 7 2 8 4 6 2	46

Poznámka.

Existuje viac než jedno efektívne riešenie. To očividnejšie je menej efektívne.

2. O Zázvorovom pive III alebo Ojojoj, Voronoi

15 bodov

Zázvorník Zachariáš sa živí distribúciou zázvorového piva Zázvorové pivo III $^{\mathrm{TM}}$. Už to raz chodí tak, že ženy sú večne doma a muži v krčme[1] a tak Zachariáš dňom i nocou bohatne.

[1] akákoľvek podobnosť s realitou je čisto náhodná

Rozvoz Zázvorového piva III ^{TM} funguje nasledovne: V niektorých mestách sú sklady s prakticky neobmedzenými zásobami, a v každom sklade je jeden závozník. Vždy, keď niekam treba dodať nové sudy piva, Zachariáš zavolá niektorému závozníkovi, ten naloží sudy do auta a najkratšou cestou ich na miesto určenia dovezie.

Má to však jeden háčik. Zachariáš často nevie, ktorého závozníka poslať doručiť pivo. A tak sa bežne stane, že niektorý iný závozník by to mal na dané miesto bližšie ako ten, ktorého Zachariáš poslal. A s rastúcimi cenami benzínu je toto čím ďalej, tým vážnejší problém.

Preto sa Zachariáš rozhodol, že úplne nutne potrebuje mapu, z ktorej by vedel pre ľubovoľné miesto (v meste alebo aj na niektorej ceste) rýchlo povedať, ktorého závozníka tam poslať.

Úloha: Na vstupe je daný počet miest n, počet závozníkov z a počet ciest m. Mestá sú očíslované 1 až n tak, že závozníci sú v mestách 1 až z.

Ďalej nasleduje zoznam m ciest. Trojica (u, v, d) znamená, že medzi mestami u a v vedie cesta dĺžky d. Úlohou je pre každú cestu vypísať, ktorý závozník to má ku ktorej jej časti najbližšie. (Pozri si príklad vstupu a výstupu.)

Príklad:

Vstup:	**Výstup:**
$n = 4$, $z = 3$, $m = 4$	cesta $(1, 4)$: hocikam na tejto ceste to má najbližšie závozník č. 1
$(1, 4, 1)$	cesta $(1, 2)$: do polovice cesty je najbližšie č. 1, zvyšok cesty je to č. 2
$(1, 2, 4)$	cesta $(4, 2)$: do vzdialenosti 1 od mesta 4 je najbližšie č. 1, potom č. 2
$(4, 2, 3)$	cesta $(4, 3)$: do vzdialenosti 2 od 4 je najbližšie č. 1, potom č. 3
$(3, 4, 5)$	

3. Odpovedz, je tam hrana? 15 bodov

Vedcom z IBM[2] už v minulosti pár krát úspešne pomohli riešitelia KSP. A aj tentoraz potrebujú vašu pomoc. Dostali za úlohu vytvoriť program, ktorý dostane na vstupe najprv popis nejakého planárneho grafu a potom bude musieť interaktívne odpovedať na otázky, či je alebo nie je medzi danými vrcholmi hrana.

Pre tých, čo nevedia, čo je planárny graf: Pod pojmom graf si môžeme predstaviť nejakú množinu bodov nakreslených na papieri, z ktorých niektoré dvojice bodov sú pospájame čiarami. Bodom budeme hovoriť vrcholy a čiaram, ktoré ich spájajú, hrany. Medzi každou dvojicou vrcholov vedie najviac jedna hrana. Planárny graf je taký graf, ktorý vieme nakresliť do roviny (na hárok papiera) tak, že ľubovolné hrany sa môžu pretnúť iba v svojom spoločnom konci. T.j. jediným možným prienikom dvoch hrán je nejaký vrchol grafu, z ktorého obe vychádzajú. Príkladom neplanárneho grafu je napríklad graf pozostávajúci z piatich vrcholov a desiatich hrán – každá dvojica vrcholov je spojená hranou. Tento graf sa nedá nakresliť do roviny tak, aby sa nejaké dve hrany „v strede“ nepretli.

Úloha:

Na vstupe váš program dostane počet vrcholov N a počet hrán M v grafe. Vrcholy sú očíslované od 1 po N. Ďalej nasleduje M dvojíc čísel vrcholov, ktoré sú spojené hranou. Medzi každou dvojicou vrcholov vedie najviac jedna hrana. Môžete predpokladať, že graf na vstupe je planárny.

Potom bude váš program interaktívne odpovedať na otázky, či je alebo nie je medzi danou dvojicou vrcholov hrana. Teda v každej otázke sa ho opýtame na dvojicu čísel a má odpovedať „ÁNO“ alebo „NIE“ podľa toho, či sa táto dvojica nachádzala v popise grafu (na poradí prvkov v dvojici samozrejme nezáleží).

[2] Institute of Black Magic

Váš program má odpovedať na každú otázku skôr, ako dostane ďalšiu. Na jednotlivé otázky sa môžeme opýtať aj viackrát, teda počet otázok nie je dopredu známy a preto musí byť váš program schopný pracovať do nekonečna.

Vedcom z IBM sa ešte podarilo objaviť dôležitú vlastnosť planárných grafov, s ktorou sa s vami radi podelia: Počet hrán v každom planárnom grafe s aspoň tromi vrcholmi je maximálne $3N - 6$, kde N je počet vrcholov grafu.

Kritériá hodnotenia vašich programov

- Najdôležitejšie je, že váš program by mal používať len lineárne veľa pamäte od veľkosti vstupného grafu. (Ak túto požiadavku nesplní, dostane veľmi málo bodov.)
- Na otázky by mal odpovedať čo najrýchlejšie.
- Pri/tesne po načítaní grafu môže váš program stráviť nejaký čas predpočítavaním nejakých údajov. Ak máme dva programy, ktoré na otázky odpovedajú rovnako rýchlo, lepší je ten, ktorý potrebuje na predpočítanie kratší čas.

Príklad:

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

Komunikácia:
```
? 2 1
ANO
? 1 4
NIE
? 5 3
NIE
? 2 3
ANO
```

4. O skokanoch

15 bodov

Exotický šach je hra, ktorá funguje podobne ako obyčajný, len sa napríklad použijú iné figúrky. Jedným typom figúrok sú *skokani*. Skokan sa dá popísať konečnou množinou povolených skokov, teda políčok, na ktoré vie zo svojho políčka skočiť bez ohľadu na to, aké figúrky sú „po ceste“. Každý skok vieme popísať dvojicou čísel $[x, y]$, ktoré znamenajú „posuň sa o x políčok doprava a o y dopredu“.

Požadujeme však, aby množina možných skokov bola symetrická so stredom symetrie v mieste, kde figúrka stojí – teda ak vie skokan niekam skočiť, musí vedieť v nasledujúcom ťahu skočiť späť.

Príkladom skokana je klasická šachová figúrka *jazdec*. Tú môžeme popísať množinou povolených skokov: $[1, 2]$, $[1, -2]$, $[2, 1]$, $[2, -1]$ a skoky k nim opačné.

Predstavme si teraz, že skokana položíme na nekonečnú štvorčekovú sieť na políčko so súradnicami $[0, 0]$. Ofarbime na zeleno políčka, kam sa vie konečným počtom skokov dostať. Tieto políčka nazveme jeho *teritóriom*.

Úloha: Na vstupe sú popisy dvoch skokanov. Zistite, či majú rovnaké teritóriá.

Príklad:

Vstup:
prvý skokan: $[1, 2]$, $[1, -2]$, $[2, 1]$, $[2, -1]$
druhý skokan: $[1, 0]$, $[0, 1]$, $[1, 1]$, $[1, -1]$

Výstup:
ANO
(Prvý skokan je jazdec, druhý je kráľ, teritóriom oboch je celá rovina.)

Vstup:
prvý skokan: $[1, 2]$, $[2, 1]$
druhý skokan: $[3, 0]$, $[0, 4]$

Výstup:
NIE
(Prvý sa vie dostať na $[3, 3]$, druhý nie.)

5. Optimálna cestná sieť

15 bodov

Absurdistan je široká ďaleká rovina a v tejto rovine leží N miest. Donedávna si tam tieto mestá len tak ležali a obyvatelia sa medzi nimi prevážali na koňoch. Až prišiel osudný deň, keď si šejk Al-adár dal doviezť z Európy auto. Teraz si preň dal postaviť cestnú sieť, po ktorej by sa dalo dostať z ľubovoľného mesta do ľubovoľného iného. Ako to ale chodí, miestni obyvatelia sú leniví, preto chcú túto cestnú sieť postaviť čo najkratšiu.

To ale nie je len tak ľahké, ako sa možno na prvý pohľad zdá. Zamyslime sa napríklad nad situáciou, keď chceme spojiť tri mestá, ktoré sú vo vrcholoch rovnostranného trojuholníka. Najlepším riešením v tomto prípade je spraviť v ťažisku trojuholníka križovatku a postaviť cesty z nej do všetkých troch vrcholov.

Úloha:

Vašou úlohou bude nájsť čo najlepšie (nie nutne optimálne!) riešenie pre niekoľko konkrétnych vstupov. Na adrese „`http://ksp.sk/~liahen/data/trees.zip`“ si môžete tieto vstupy stiahnuť.

Formát každého vstupného súboru je nasledujúci:

Na prvom riadku je počet miest N ($1 \leq N \leq 1\,000$). Nasleduje N riadkov, v i-tom z nich sú dve celé čísla X_i, Y_i (medzi $-300\,000$ a $300\,000$, vrátane) – súradnice jedného z miest. Mestá sú očíslované od 1 do N v poradí, v akom sú zadané vo vstupnom súbore.

Formát výstupného súboru by mal byť nasledujúci:

V prvom riadku bude celé číslo M ($0 \leq M \leq N$) – počet nových križovatiek, ktoré chceš postaviť. Nasleduje M riadkov, každý z nich obsahuje dve reálne čísla (medzi $-300\,000$ a $300\,000$, vrátane, v ľubovoľnom štandardnom formáte) – súradnice jednej križovatky. Križovatky očíslujeme od $N + 1$ do $N + M$ v poradí, v akom sú uvedené v tvojom výstupnom súbore.

Zvyšok výstupného súboru tvorí $N + M - 1$ riadkov. Každý z nich obsahuje dve čísla miest/križovatiek, ktoré chceš v svojej cestnej sieti spojiť priamou cestou.

(Postavené cesty musia vytvoriť súvislú cestnú sieť. V optimálnom riešení sa zjavne žiadne dve cesty nepretínajú. Ak sa v tvojom riešení nejaké cesty pretínať budú, považujeme toto miesto za mimoúrovňovú križovatku, teda auto tam nemôže prejsť z jednej cesty na druhú.)

V ideálnom prípade odovzdajte svoje riešenia priamo v Programátorskej liahni, kde túto úlohu nájdete (v piatej sade, pod rovnakým názvom). Každému sa bude do KSP hodnotiť najlepší dosiahnutý výsledok pred uplynutím termínu série. Pokiaľ niekomu tento spôsob odovzdávania riešení nevyhovuje, môže ich poslať mailom na adresu `ksp` zavináč `ksp.sk`.

Zdôrazňujeme, že každé zlepšenie riešenia je dôležité. Čím bližšie k optimálnemu riešeniu sa dostanete, tým bude zlepšovanie riešení ťažšie, ale každý dosiahnutý úspech cennejší.

Príklad:

Vstup:	**Výstup:**
3	1
0 0	500 288.67
1000 0	1 4
500 866	2 4
	4 3