Korešpondenčný seminár z programovania
XXI. ročník, 2003/04
Katedra vyučovania informatiky FMFI UK,
Mlynská Dolina, 842 48 Bratislava

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

Príklady 1. kola letnej časti

Milí riešitelia!

Vítame vás v letnej časti KSP, ktorá práve pomaly ale isto začína. A veríme, že podobne, ako sa dočkáme jari, dočkáme sa aj kopy riešení od vás. Neváhajte a zapojte sa – prvých niekoľko riešiteľov čakajú vecné ceny, s najlepšími riešiteľmi sa stretneme na skvelom týždňovom sústredení a ak aj neuspejete, aspoň sa naučíte mnoho vecí, ktoré sa vám ešte v živote zídu. Zapojiť sa môžu aj tí, ktorí KSP v zime neriešili, nebudú nijako znevýhodnení.

Riešenia tohto kola nám posielajte do **15. marca 2004** a nezabudnite priložiť pár známok v hodnote okolo 15 korún, nech vám ich máme za čo poslať späť opravené.

Ani fakír nemá na ružiach ustlané!

KSPáci

1. O matematikoch

15 bodov

Matematici si radi dávajú matematické hádanky. Ak matematik zadá druhému nejakú matematickú úlohu a ten ju do týždňa nevyrieši, tak musí pozvať svojho vyzývateľa na dobrú hostinu. (To je vecou cti.)[1] Nuž a jeden úbohý matematik rieši jednu zapeklitú úlohu už skoro týždeň a nevie ju vyriešiť. Pomôžte mu!

Úloha: Je dané N a množina obsahujúca N (rôznych) čísel. Nájdite štyri čísla a, b, c, d z tejto množiny (nie nutne rôzne) tak, aby platilo $a + b + c = d$. Prípadne vypíšte, že také štyri čísla v množine nie sú.

Príklad:

Vstup:
$N = 9$
čísla: -15, 1, 4, 10, 2, 9, 7, 3, -1

Výstup:
$1 + 1 + 7 = 9$

2. Oooooooo Bisťu Blabla Bum

15 bodov

Bum Bum. „Ja ti ukážem!“ Bum. Trésk. Chrap. Rozbitý kvetináč. Zlomené dvere. „Aúúúúúú“. Spadnutá polička. „Ty pablb škaredý!“ Pred dvojicou stánkov s párkami sa mlátili dvaja predavači z nich. Konkurenčný boj nadobúdal na tvrdosti. „Presťahuj si tú tvoju krabicu!“ „To ty sa prac (prásk), ja som tu bol prvý!“ Obďaleč postával dav ľudí – niektorí zvedaví, niektorí si jednoducho chceli kúpiť párky, no nebolo u koho.

„Takto to ďalej nejde!“ usúdil radný pán Richard (ktorý práve stál v spomínanom dave a patril do druhej skupiny ľudí). „Vydáme nariadenie, že stánky s párkami nesmú stáť príliš blízko seba!“ A stalo sa. Predavači s párkami boli zrazu donútení spolupracovať a vyriešiť dilemu – kde majú postaviť stánky, aby neporušili podmienky mestskej rady a ušlo sa miesto každému z nich?

[1] Ešte pred 200 rokmi tieto bývali tieto súboje vcelku bežné.

A nebojte sa, nedošlo k zmene ich charakteru... Len čo im pomôžete vyriešiť túto úlohu, pobijú sa o to, ktorí z nich budú mať stánky na lukratívnejších miestach.

Úloha: Mesto sa skladá z križovatiek a ulíc, ktoré medzi nimi vedú. Keďže mesto je malé, medzi každými dvoma križovatkami sa dá po uliciach prejsť len jedným spôsobom – matematik by povedal, že cestná sieť mesta je strom. Stavať stánky na križovatkách sa nesmie. Ak na niektorej ulici postavíme stánok s párkami, nemôžeme už žiadny iný postaviť ani na tejto ulici, ani na žiadnej z tých, ktoré s ňou susedia. (Dve ulice sú susedné, ak majú spoločnú križovatku.) Váš program by mal v meste rozmiestniť najväčší možný počet stánkov s párkami.

Príklad:

Vstup:	**Výstup:**
Počet križovatiek: $N = 9$ Ulice: 1-9, 1-2, 2-3, 3-4, 4-5, 4-6, 4-7, 4-8 *(Križovatky sú očíslované od 1 do N, každá ulica je zadaná číslami križovatiek, ktoré spája.)*	Stánky treba dať na ulice: 1-9, 2-3, 4-6

3. O pánoch Hamiltonovi a di Rakovi

15 bodov

Pán Hamilton je obchodný cestujúci. Má taký veľký hnedý kufor a v ňom kopec vecí, ktoré by si niekto mohol kúpiť. A s týmto kufrom cestuje po svete a koho stretne, tomu ponúka čokoľvek, čo vo vnútri nájde.

Pán Hamilton má však momentálne práce vyše hlavy, a preto si vzal dovolenku. So svojim starým známym priateľom pánom di Rakom sa spoločne vybrali navštíviť isté tichomorské súostrovie.[2] V súostroví sa nachádza N ostrovov. Tam podnikaví domorodci poskytujú plavbu člnom medzi niektorými ostrovmi. Pán Hamilton je značne poznačený svojou prácou a tak vás iste neprekvapí, že by chcel navštíviť každý ostrov práve raz. Teraz je však na dovolenke, preto mu nezáleží, ako dlho bude trvať obhliadnutie ostrovov.

Pán Hamilton má pochybnosti, či vôbec existuje spôsob, ako navštíviť všetky ostrovy a každý práve raz. Jeho priateľ pán di Rak ho ubezpečuje, že určite áno. Veď z každého ostrova ich podnikaví domorodci môžu dopraviť na viac ako $\frac{N}{2}$ ďalších ostrovov. Sami uznajte, to predsa musí ísť. Pán Hamilton je však stále nedôverčivý. Pomôžte mu a zistite, či je možné stráviť dovolenku podľa jeho predstáv.

Úloha: Na vstupe je dané N – počet ostrovov v súostroví. Ostrovy sú pre jednoduchosť očíslované 1 až N. Ďalej nasleduje zoznam dvojíc ostrovov, medzi každou dvojicou premáva nejaký podnikavý domorodec na člne a pýta sa turistov, či nechcú odviezť. Vašou úlohou je nájsť (ak existuje) plán cesty, ako môže pán Hamilton navštíviť každý ostrov práve raz a na konci sa vrátiť na ostrov, z ktorého začal. Ak takýto plán neexistuje, vypíšte o tom príslušnú správu. Predpokladajte, že z každého ostrova sa dá dostať na viac ako $\frac{N}{2}$ iných ostrovov.

Príklad:

Vstup:	**Výstup:**
N = 5 1 2 1 4 1 5 2 3 2 4 2 5 3 4 3 5	Ostrovy navštívi v poradí: $1, 5, 2, 3, 4$.

[2] ktorého meno neprezradíme, lebo to by ich tam potom každý vyrušoval

4. Objemné teleso

15 bodov

Predstavte si kôpku rovnako veľkých kociek. Predstavte si, že z nich pozliepame súvislé teleso. Každé dve kocky, ktoré susedia, sa musia dotýkať celou stenou. Predstavte si niekoľko takýchto telies. Môžu pokojne byť deravé alebo duté. Teraz si predstavte, že sme povrch nášho telesa polepili niekoľkými kusmi tvrdého papiera. Zjavne sa zaobídeme bez takých kusov, ktoré sú deravé – každý deravý kus môžeme rozdeliť na niekoľko nederavých. Takže povrch každého telesa z kociek môžeme popísať pomocou niekoľkých papierových mnohouholníkov.

Úloha: Na vstupe je zoznam papierových mnohouholníkov, z ktorých sa skladá povrch daného telesa. Každý mnohouholník leží v rovine, ktorá je rovnobežná s rovinou určenou osami **xy**, **xz** alebo **yz**. Každá hrana každého mnohouholníka je rovnobežná s niektorou zo súradnicových osí. Súradnice vrcholov sú celé čísla. Vašou úlohou je napísať program, ktorý spočíta, z koľkých kociek sa zadané teleso skladá.

Príklad:

Vstup:

stien: 12
vrcholov: 4 vrcholy: (10,10,10), (10,10,20), (10,20,20), (10,20,10)
vrcholov: 4 vrcholy: (20,10,10), (20,10,20), (20,20,20), (20,20,10)
vrcholov: 4 vrcholy: (10,10,10), (10,10,20), (20,10,20), (20,10,10)
vrcholov: 4 vrcholy: (10,20,10), (10,20,20), (20,20,20), (20,20,10)
vrcholov: 4 vrcholy: (10,10,10), (10,20,10), (20,20,10), (20,10,10)
vrcholov: 4 vrcholy: (10,10,20), (10,20,20), (20,20,20), (20,10,20)
vrcholov: 4 vrcholy: (14,14,14), (14,14,16), (14,16,16), (14,16,14)
vrcholov: 4 vrcholy: (16,14,14), (16,14,16), (16,16,16), (16,16,14)
vrcholov: 4 vrcholy: (14,14,14), (14,14,16), (16,14,16), (16,14,14)
vrcholov: 4 vrcholy: (14,16,14), (14,16,16), (16,16,16), (16,16,14)
vrcholov: 4 vrcholy: (14,14,14), (14,16,14), (16,16,14), (16,14,14)
vrcholov: 4 vrcholy: (14,14,16), (14,16,16), (16,16,16), (16,14,16)

Výstup:

objem: 992

(Kocka $10 \times 10 \times 10$ *s dierou* $2 \times 2 \times 2$*)*

5. O mravcoch III

15 bodov

Určite si ešte pamätáte, že v prvej sérii dostali mravce na narodeniny kráľovnej počítače. Ak ste túto sériu neriešili, nezúfajte, všetko dôležité si ešte raz pripomenieme. Tieto počítače sú rovnaké a je ich veľmi veľa. Vašou úlohou bude napísať program, ktorý keď nahráme na všetky počítače a naraz ho na všetkých spustíme, niečo zmysluplné spočíta.

Ak si pamätáte, ako sa mravčie počítače programujú, môžete preskočiť priamo k zadaniu úlohy. Pre ostatných teraz bližšie popíšeme, ako vyzerá programovací jazyk, v ktorom sa programujú mravčie počítače. Program budeme písať v jazyku podobnom jazyku Pascal. Premenné použité v programe budú lokálne, t.j. každý počítač bude mať vlastné premenné. Jedinou výnimkou je globálne pole celých čísel `Mem[]`. Toto pole je indexované od 0 a neobmedzene veľké. Na začiatku je v ňom (väčšinou v prvých niekoľko políčkach) uložený vstup úlohy, ostatné jeho políčka obsahujú nuly. Na konci v ňom má ostať uložený výsledok úlohy.

Programy budú spustené naraz na všetkých počítačoch. Počítače budú očíslované od 1 do K. Výpočet bude prebiehať v taktoch. Všetky počítače sú rovnako rýchle, v každom takte každý z nich vykoná práve 1 inštrukciu.

Na začiatku programu má byť deklarácia lokálnych premenných, nasledujú samotné inštrukcie. V programe musí byť každá inštrukcia uvedená na samostatnom riadku (teda v jednom takte vykoná každý počítač práve jeden riadok programu). Riadok môžeme ukončiť bodkočiarkou, ňou zároveň začína komentár. Povolené inštrukcie sú:

- priradenie
- prázdna inštrukcia **nop**
- inštrukcia ukončenia výpočtu **halt**
- príkaz skoku **goto** *návestie*

• podmienený príkaz **if** *podmienka* **then** *inštrukcia* (**else** *inštrukcia*)

Pravú stranu priradenia môže predstavovať ľubovoľný aritmetický výraz, podmienkou môže byť ľubovoľný logický výraz. Okrem premenných a konštánt môžu tieto výrazy obsahovať volanie funkcie **cpuid**, ktorá vráti číslo počítača, na ktorom program beží. (Všimnite si, že nemáte k dispozícii kľúčové slová **begin** a **end**.) Návestia sa definujú rovnako ako v Pascale: napísaním *názov:* pred príslušný riadok.

Zostáva vyriešiť prácu so spoločnou pamäťou. Môže sa stať, že niekoľko počítačov chce v tom istom takte čítať z, resp. zapisovať na to isté pamäťové miesto. V takomto prípade najskôr prebehne čítanie, potom zápis, Čítať jedno políčko globálnej pamäte môže naraz ľubovoľne veľa počítačov, ale zapisovať dovolíme len jednému. Ak by naraz chcelo na to isté miesto zapisovať viac počítačov, program vráti chybu. (T.j. tomuto sa pri písaní programu treba vyhnúť.)

Ukážeme si teraz, v čom je sila mravčích počítačov. Na klasickom počítači potrebujeme na nájdenie najväčšieho z daných N čísel čas $O(N)$. Pozrime sa, ako rýchlo ho vedia nájsť mravce. Nech teda v `Mem[0]` je uložené N a v `Mem[1]` až `Mem[N]` jednotlivé čísla. Spustime na N počítačoch nasledovný program:

```
  var N,num1,num2 : integer;
  N:=Mem[0];
1: if (2*cpuid-1 > N) then halt;
  num1:=Mem[2*cpuid-1];
  if (2*cpuid <= N) then num2:=Mem[2*cpuid] else num2:=num1;
  if (num1<num2) then Mem[cpuid]:=num1 else Mem[cpuid]:=num2;
  N:=(N+1) div 2;
  if (N=1) then halt;
  goto 1
```

Po skončení bude v `Mem[1]` uložené najväčšie spomedzi daných čísel. Prečo? Počas prvého prechodu cyklom sme rozdelili zadané čísla na dvojice, z každej sme vybrali to väčšie a následne sme „víťazov“ uložili na políčka $1 \ldots \lceil N/2 \rceil$ – tým sme vlastne dostali pôvodnú úlohu s približne polovičným počtom čísel, to celé v konštantnom čase! Preto náš program naozaj nájde maximum spomedzi daných N čísel, a to v čase $O(\log N)$.

Úloha: V poli `Mem[]` je uložená postupnosť **navzájom rôznych** celých čísel. (V `Mem[0]` je uložená jej dĺžka N, v `Mem[1]` až `Mem[N]` sú jednotlivé prvky postupnosti.) Váš program by mal túto postupnosť utriediť.

Teda ak je na vstupe postupnosť (5,3,7,1,12,9),[3] na výstupe by mala byť postupnosť (1,3,5,7,9,12).

Môžete uviesť, koľko počítačov má byť na začiatku spustených v závislosti od N, počet počítačov však smie od N závisieť nanajvýš polynomiálne (teda $N^4 + 7$ počítačov je v poriadku, ale 2^N už nie.)

Prvoradým kritériom hodnotenia bude *čas výpočtu vášho programu*, nasleduje *použitá pamäť* (súčet lokálnych pamätí spustených počítačov a počtu použitých políčok globálnej pamäte) a *počet použitých počítačov*.

Možno hinty: Ktoré klasické algoritmy triedenia sa dajú ako dobre paralelizovať? Pomohlo by nám, keby sme mohli použiť až $N!$ počítačov? A treba ich vlastne až tak veľa?

[3] Teda v `Mem[0]` je 6 – dĺžka postupnosti, v `Mem[1]` je 5, v `Mem[2]` je 3, atď.