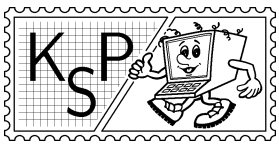


Korešpondenčný seminár z programovania
XIX. ročník, 2001/2002

Katedra vyučovania informatiky FMFI UK,
Mlynská Dolina, 842 48 Bratislava

KSP finančne podporuje nadácia Open Society Fund Bratislava
IUVENTA – zariadenie pre voľný čas detí, mládeže a dospelých
MICROSTEP spol. s.r.o.

Propozície

Korešpondenčný seminár z programovania (KSP) je súťaž programátorov – stredoškolákov. Jej cieľom je zdokonaliť žiakov, ktorí už vedia programovať alebo sa práve programovať učia, v programovaní a v algoritmickom myslení.

Kategórie: Korešpondenčný seminár z programovania pozostáva z dvoch kategórií – **KSP a KSP-Z**. Kategória KSP-Z je určená pre začínajúcich riešiteľov. V tomto ročníku sa jej môže zúčastniť každý stredoškolák, ktorý v žiadnom z predchádzajúcich ročníkov v kategórii KSP nezískal za celý ročník viac ako **30 bodov**, v kategórii KSP-Z viac ako **90 bodov**, a nezúčastnil sa sústredenia KSP. Kategórie KSP sa môžu zúčastniť všetci stredoškoláci bez obmedzenia (aj riešitelia KSP-Z). Obzvlášť radi privítame účasť dievčat.

Sústredenia: Najlepších riešiteľov oboch kategórií pozývame každoročne na dve týždenné sústredenia (pokiaľ sa ich podarí zorganizovať). Na jesenné sústredenie nepozývame riešiteľov, ktorí už ukončili strednú školu. Pokiaľ viete o niekom, kto by mohol tieto sústredenia čo i len čiastočne sponzorovať, dajte nám o ňom (a prípadne jemu o nás) vedieť.

Organizácia súťaže: Kategória KSP aj kategória KSP-Z sú organizované v štyroch kolách, ktoré sú rozdelené na **dva samostatné polroky po dvoch kolách**. Na jarné sústredenie teda budú pozývaní najlepší riešitelia podľa počtu bodov za 1. a 2. kolo zimnej časti a na jesenné sústredenie podľa počtu bodov za 1. a 2. kolo letnej časti. Každé kolo obsahuje päť úloh. Príklady každého kola (nie nutne všetky) je treba vyriešiť a do určeného termínu poslať na našu adresu. Riešenia odoslané po tomto termíne nemusia byť hodnotené, prípadne môžu byť riešiteľovi strhnuté body za neskoré odoslanie riešenia. Riešiteľom, ktorí sa zúčastnili niektorého kola súťaže, pošleme zadania ďalšieho kola, výsledkovú listinu a komentáre k riešeniam. Ďalšieho kola sa však môžu zúčastniť aj tí žiaci, ktorí sa predchádzajúceho kola nezúčastnili.

Požiadavky na riešenia: Základom riešenia každého príkladu (ak nie je v zadaní uvedené inak) je **listing programu** v ľubovoľnom vyššom programovacom jazyku (najlepšie Pascal, C/C++). Na programe nás zaujíma najmä jeho korektnosť (dáva program pre každý vstup správny výsledok?) a efektivita (koľko času a pamäte potrebuje na spracovanie vstupu určitej veľkosti). Dôraz kladťe na algoritmickú časť programu, pri načítavaní vstupu a vypisovaní výstup nemusíte úplne presne dodržiavať formát zo zadania. Rovnako dôležitou súčasťou riešenia je **slovný popis** riešenia. Slovný popis by mal obsahovať popis použitého algoritmu, prípadne popis kľúčových dátových štruktúr. Mal by byť natoľko jasný a zrozumiteľný, aby bolo podľa neho možné napísať program rovnako efektívny, ako ten váš. Ďalej vyžadujeme **zdôvodnenie** (dôkaz) **správnosti** použitého algoritmu a odhad časovej a pamäťovej zložitosti algoritmu (t.j. koľko času a pamäte potrebuje váš program v závislosti od veľkosti vstupných dát).

Formálna úprava: **Diskety nám neposielajte.** Prvý list každého riešenia má obsahovať hlavičku, v ktorej je meno riešiteľa, číslo príkladu, kategória, počet listov príkladu a škola. Ďalšie listy obsahujú meno, číslo príkladu a číslo listu. Je dobré zopnúť listy patriace k jednému príkladu, aby sa nestratili. Nezopínajte listy patriace rôznym príkladom a už vôbec nepíšte riešenia viacerých príkladov na ten istý list papiera.

Bodovanie: Maximálne počty bodov pre jednotlivé príklady sa v zadaniach neuvádzajú, pohybujú sa od 10 do 25 bodov podľa náročnosti príkladu. KSP je súťaž jednotlivcov, preto ak sa vyskytnú „kolektívne riešenia“, bude za ne udelených patrične menej bodov.

Evidenčný lístok: Každý súťažiaci musí poslať spolu s riešeniami prvého kola vyplnený evidenčný lístok. Veľmi dôležitou informáciou na tomto lístku je adresa pre korešpondenciu. Na túto adresu budeme posielať opravené riešenia a zadania ďalších kôl. Ak táto adresa je iná, ako vaša domáca adresa (ak bývate napr. na internáte), napíšte nám aj svoju domácu adresu, aby sme vám mohli písať aj cez prázdniny. Nezabudnite tiež uviesť ročník štúdia a označiť kategórie, ktorých sa chcete zúčastniť. Tí z vás, ktorí máte prístup k Internetu môžete evidenčný lístok vyplniť na adrese http://www.ksp.sk/ksp/prihlaska.php.

Poštovné náklady: Žiaľ nie je v našich silách hradiť všetky výdavky na poštovné (rozposielanie zadaní, riešení a vzorových riešení). Preto sme zaviedli „účastnícky poplatok“. Účastnícky poplatok je **15,- Sk** za každú sériu, ktorej sa riešiteľ zúčastní a posiela sa spolu **s riešeniami úloh formou poštových známok v rozumných hodnotách, napr. 2,- a 5,- Sk.** Účastníkom, ktorí nám nepošlú známky, hrozí, že od nás nedostanú žiadnu poštu.

Tešíme sa na vaše riešenia a prajeme vám veľa dobrých nápadov

KSPáci

Adresa: KSP
Katedra vyučovania informatiky FMFI UK
Mlynská dolina
842 48 Bratislava 4
E-mail: `ksp@ksp.sk`
URL: `http://www.ksp.sk/ksp`
Telefón: 02 / 60 295 210

Mailing list eKSPres

Mailing list eKSPres je určený riešiteľom a opravovateľom KSP, ako aj najrôznejším priaznivcom seminára. Dá sa na ňom dozvedieť o rôznych akciách organizovaných vedúcimi KSP (napríklad splav), diskutovať o zaujímavých problémoch (s výnimkou príkladov aktuálnej série KSP!) a pod.

- Ak sa chcete doňho prihlásiť, musíte mať e-mailovú adresu. Z nej pošlite na adresu `eKSPres-request@st.fmph.uniba.sk` mail obsahujúci jediné slovo `SUBSCRIBE` v tele správy. Od tohoto okamihu vám budú na vašu adresu chodiť všetky správy z mailing listu eKSPres.
- Ak chcete ostatným poslať mail, pošlite mail na adresu `eKSPres@st.fmph.uniba.sk`. Váš mail príde všetkým členom eKSPres-u, preto sa pred jeho odoslaním najprv zamyslite nad jeho vhodnosťou.

Korešpondenčný seminár z programovania
XIX. ročník, 2001/2002
Katedra vyučovania informatiky FMFI UK,
Mlynská Dolina, 842 48 Bratislava

KSP finančne podporuje nadácia Open Society Fund Bratislava
IUVENTA – zariadenie pre voľný čas detí, mládeže a dospelých
MICROSTEP spol. s.r.o.

Príklady 1. kola zimnej časti

Milé naše riešiteľky, mílí naši riešitelia,

dostávajú sa vám do rúk zadania 19. ročníka KSP. Prečítajte si pozorne propozície súťaže a hor sa riešiť. V prípade, že riešite KSP po prvý raz, odporúčame vám začať s kategóriou KSP-Z. Svoje riešenia nám posielajte najneskôr do **15. októbra**. Nezabudnite k nim priložiť známky v hodnote 15,- Sk.

Kto chce kam, pomôžme mu tam.

KSPáci

1. O výsledkovej listine

Aj tento rok sa naša výprava šťastne vrátila z Medzinárodnej olympiády v informatike (IOI), ba aj nejaké medaily si priviezli. No a poznáte Mariána. Nie aby si po príchode domov ľahol spokojne spať, sadol si za počítač a začal vyrábať štatistiky. Koľkí súťažiaci mali za niektorý príklad plný počet bodov, ako dopadli jednotlivé krajiny podľa počtu medailí, ako dopadli podľa súčtu bodov súťažiacich... Tu sa zarazil. Organizátori totiž zverejnili len body lepšej polovice súťažiacich, takže u niektorých krajín mu chýbali počty bodov niekoľkých súťažiacich. Tak spravil aspoň poradie podľa bodov, ktoré mal k dispozícii a ku každej krajine si poznačil, koľko mu z nej ešte chýba súťažiacich (z každej krajiny sú na IOI najviac štyria súťažiaci). Teraz by chcel vedieť, ako najlepšie a najhoršie môže byť každá krajina umiestnená v poradí podľa súčtu bodov všetkých súťažiacich. To je ale úloha pre počítač, a keďže Marián je na zaslúženom odpočinku, ostalo to na vás.

Úloha: Vašou úlohou je napísať program, ktorý dostane na vstupe počet krajín, ich zoznam, pre každú krajinu súčet bodov súťažiacich, ktorých body Marián vie a počet súťažiacich z nej, ktorých body nevie. Tento zoznam bude zotriedený podľa počtu známych bodov. Okrem toho na vstupe dostane najmenší zverejnený počet bodov (od ktorého má každý z chýbajúcich súťažiacich určite menej). Váš program by mal pre každú krajinu vypísať, ako najlepšie a ako najhoršie môže byť umiestnená v poradí podľa súčtu bodov všetkých jej súťažiacich.

Príklad:

Vstup:

```
Slovensko 1470 0
Singapur 1350 0
Rusko 1150 1
Čína 850 2
250
```

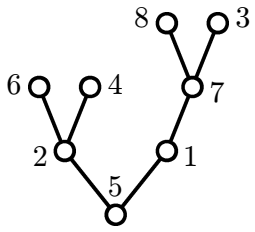
Výstup:

Slovensko:	1. – 1.
Singapur:	2. – 3.
Rusko:	2. – 4.
Čína:	3. – 4.

(Čína môže mať dokopy najviac 1348 bodov, preto je určite za Singapurom, a teda najlepšie tretia. Keby chýbajúci Rus získal viac ako 200 bodov, je Rusko pred Singapurom aj Čínou. A tak ďalej.)

2. O čarovnom strome

Kde bolo, tam bolo, bolo raz jedno kráľovstvo, kde múdry Kráľovič panoval. Panoval on veru dobre, a preto sa ho ani nik nesnažil z trónu zosadiť.



Jedného dňa dopočul sa múdry Kráľovič o akomsi čarovnom strome kdesi v susednom kráľovstve. Nemal to byť len tak obyčajný strom, teda aspoň podľa tých rečí, čo sa sem doniesli. Všade, kde sa strom vetvil, sa vetvil na 2 vetvy, jedna rástla doľava, druhá doprava. Občas sa stalo aj to, že jedna z vetví sa odlomila a potom ostala len tá druhá. Každá nová vetva sa mohla ďalej vetviť, až časom narástol celý strom. Navyše, tento čarovný strom má v každom rozvetvení napísané nejaké číslo, a taktiež má podobné čísla na listoch (list je tam, kde končí vetvička). A ešte sa povráva, že žiadne z týchto čísel nie sú rovnaké a že sú tam všetky čísla začínajúc od jednotky. Taký strom môže vyzerať napríklad ako ten na obrázku.

Nebol by to múdry Kráľovič, keby nechcel čo najskôr vedieť, ako presne ten čarovný strom vyzerá. Rozhodol sa preto svojich troch synov na prieskum do susedného kráľovstva vyslať. Aby tam však nešli s rovnakou úlohou, múdry Kráľovič riekol: »*Ty, najstarší syn môj, prinesieš popis čarovného stromu v* Pre-Order *zápise, ty, prostredný syn môj, prinesieš* In-Order *zápis a ty, najmladší syn môj, ty dones* Post-Order *zápis čarovného stromu.*«

Pobral sa najstarší, pobral sa i prostredný syn čarovný strom navštíviť a jeho zápis získať. I najmladší sa už-už na cestu vybral, keď si to rozmyslel a povedal si, že vyčká svojich bratov a potom uvidí, čo ďalej. Vrátil sa najstarší syn z dlhej cesty a hneď od dverí otcovi hlási: *„5 2 6 4 1 7 8 3“*! Čoskoro i prostredný syn vracia sa z cesty a s podobným nadšením ohlasuje svoj zápis: *„6 2 4 5 1 8 7 3“*!

Najmladší syn, ktorý si oba zápisy pozorne zapísal, pomaly začal hovoriť: *„6. . . 4. . . 2. . . “*

Úloha: Pomôžte najmladšiemu Kráľovičovmu synovi zo zadaného Pre-Order a In-Order zápisu vytvoriť Post-Order zápis čarovného stromu.

Všetky 3 uvedené zápisy stromu sú podobné. Strom v Pre-Order zápise sa zapisuje tak, že začneme pri koreni (na obrázku je to 5), ten zapíšeme, a potom rekurzívne zapíšeme (rovnakým postupom) ľavý podstrom a potom pravý podstrom, teda *„koreň (ľavý podstrom) (pravý podstrom)“*. V našom prípade to bude vyzerať ako *„5 (2 6 4) (1 7 8 3)“*, a ak ešte vynecháme tie zátvorky, dostaneme hľadaný Pre-Order zápis. Podobne, v In-Order zápise najpr rekurzívne zapíšeme ľavý podstrom, potom zapíšeme koreň podstromu a potom rekurzívne pravý podstrom, teda *„(ľavý podstrom) koreň (pravý podstrom)“*, v našom prípade *„6 2 4 5 1 8 7 3“*, no a v Post-Order zápise najpr zapíšeme ľavý podstrom, potom pravý a až nakoniec dáme koreň podstromu, teda *„(ľavý podstrom) (pravý podstrom) koreň“* alebo v tomto prípade *„6 4 2 8 3 7 1 5“*.

Príklad:

Vstup:		**Výstup:**
Pre-Order:	5 2 6 4 1 7 8 3	6 4 2 8 3 7 1 5
In-Order:	6 2 4 5 1 8 7 3	(príklad zodpovedá obrázku zo zadania)

3. Ondrejove zápalky

Ondrej sa od mladi rád hrával so zápalkami. Staval z nich všakovaké podivuhodné umelecké dielka a rád ich rozdával svojim kamarátom. Aby tie dielka lepšie vyzerali, potreboval rôzne druhy zápaliek – s hnedou hlavičkou, so zelenou, celé červené a mnoho iných druhov.

Z dlhej chvíle si z nich začal skladať na stole mnohouholníky. Na stole má 4 druhy zápaliek - modré, zelené, červené a ružové. Všetky druhy zápaliek majú rovnakú dĺžku.

Len tak si zmyslel, že modré zápalky bude dávať len v smere zhora nadol, zelené len zľava doprava, červené len zľava dole doprava hore a ružové len zľava hore doprava dole. Teraz ho trápi, či možno zo všetkých zápaliek, ktoré má na stole, postaviť jeden veľký mnohouholník.

Úloha: Napíšte program, ktorý dostane na vstupe 4 čísla A, B, C, D – počty modrých, zelených, červených a ružových zápaliek a zistí, či sa dá z nich postaviť mnohouholník. Ak sa dá, vypíšte aj farby zápaliek v poradí, v akom sú uložené na jeho obvode. Ak je možností, ako poukladať zápalky, viacero, vypíšte ľubovoľnú z nich.

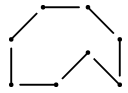
Príklad:

Vstup:
$A = 2$, $B = 2$, $C = 1$, $D = 0$

Výstup:
Z týchto zápaliek sa mnohouholník postaviť nedá.

Vstup:
$A = 2$, $B = 2$, $C = 2$, $D = 2$

Výstup:
modrá, zelená, červená, ružová,
modrá, ružová, zelená, červená



4. O menovej reforme

Na Kiribati sa tento rok chystá veľká menová reforma. Namiesto doterajších mušličiek chce vláda zaviesť mince. Prebehla veľká reklamná kampaň pod heslom „Chceme pevnejšiu menu!“, ale len tak medzi nami: hlavný dôvod je ten, že vláda už nechce, aby si ľudia chodili zbierať peniaze ráno na pláž, zle to vplýva na pracovnú morálku. Ale s takou menovou reformou to nie je len tak jednoduché.

Minister pre reformu dobre vie, že bežní občania nemajú s mincami skúsenosti. Preto keď bude mať občan zaplatiť nejakú sumu, bude ju určite platiť tak, že vždy dá najväčšiu mincu, akú môže. Napríklad mincami s hodnotami 1,3 a 5 by 7 platil ako 5+1+1. Nikoho ale nebude baviť vyťahovať zbytočne veľa mincí. Preto by bolo dobré, keby pri platení ľubovoľnej sumy by na to občania týmto postupom použili minimálny možný počet mincí.

Prvý vládny návrh boli mince s hodnotami 1, 4 a 5. Potom si ale minister uvedomil, že táto sada mincí nie je pre Kiribati vhodná. Občania by totiž napríklad 8 platili ako 5+1+1+1, a pritom to ide s menej mincami, napríklad 4+4. Skôr či neskôr by si to niekto uvedomil a začali by občianske nepokoje. A to naozaj nepotrebujú.

Nato začali ministrovi podriadení húfne podávať nové a nové návrhy sád mincí, no a chudák minister nad nimi teraz sedí a snaží sa zistiť, či je aspoň jeden z nich vhodný pre Kiribati.

Úloha: Vašou úlohou je napísať program, ktorý dostane na vstupe počet rôznych druhov mincí K a ich hodnoty $1 = a_1 < a_2 < \ldots < a_K \leq 100$ a zistí, či je táto sada mincí vhodná pre Kiribati. Pokiaľ nie, mal by nájsť aspoň jednu sumu, ktorá sa dá zaplatiť menej mincami, ako by to spravili Kiribatčania.

Príklad:

Vstup:
1 3 4

Výstup:
Nie: 6 (6 by platili ako 4+1+1 a dá sa ako 3+3.)

Vstup:
1 3 6

Výstup:
Sada je pre Kiribati vhodná.

5. O čiernych krabičkách

V softvérovom družstve SoDr sa rozhodli rozšíriť pamäť ich jediného počítača. Preto k nemu dokúpili niekoľko veľkokapacitných pamätí typu Čierna skrinka 1.0. Pamäť typu Čierna skrinka má veľmi vysokú kapacitu, má však jednu nevýhodu – nedá sa pristupovať k dátam na ľubovoľnej adrese, ale s dátami uloženými v pamäti treba pracovať iba pomocou špeciálnych príkazov.

Čierna skrinka 1.0 používa tieto tri príkazy:

- `procedure Push(dest,val);` – uloží do čiernej skrinky `dest` hodnotu `val`
- `function Pop(dest);` – vyberie z čiernej skrinky `dest` hodnotu, ktorá bola do nej vložená ako posledná. Túto hodnotu funkcia zmaže z pamäte a vráti ako návratovú hodnotu.
- `function Empty(dest);` – vráti `TRUE`, ak je čierna skrinka `dest` prázdna, inak vráti `FALSE`

Každý z týchto príkazov sa uskutočňuje v jednotkovom čase.

Príklad použitia pamäte:

Príkaz	**Návratová hodnota**	**Stav pamäte po vykonaní príkazu**
Push(a,1)		1
Push(a,3)		1,3
Empty(a)	FALSE	1,3
Pop(a)	3	1
Pop(a)	1	
Empty(a)	TRUE	

Pamäť Čierna skrinka 1.0 teda pracuje ako zásobník. V softvérovom družstve SoDr by však potrebovali pamäť, z ktorej by sa dal vyberať prvok, ktorý bol do pamäte vložený ako prvý, potrebovali by pamäť pracujúcu ako fronta. Budú preto potrebovať vašu pomoc.

Úloha: Napíšte program, ktorý bude emulovať pamäť pracujúcu ako fronta. Program má načítavať a spracovávať tieto príkazy:

- `procedure Put(val);` – uloží do pamäte hodnotu `val`
- `function Get;` – vyberie z pamäte hodnotu, ktorá bola do nej vložená ako prvá. Túto hodnotu funkcia zmaže z pamäte a vráti ju ako návratovú hodnotu.
- `function Empty;` – vráti `TRUE`, ak je pamäť prázdna, inak vráti `FALSE`

Váš program môže používať konštantný počet premenných typu Čierna skrinka 1.0. Okrem nich môže používať len konštantný počet iných premenných.

Snažte sa, aby váš program pracoval efektívne, teda aby vedel čo najrýchlejšie vedel vykonať postupnosť N príkazov a aby celkový počet dát uložených v premenných typu Čierna skrinka bol čo najmenší.

Príklad:

Vstup:	**Výstup:**
Put(1)	
Put(3)	
Empty	FALSE
Get	1
Get	3
Empty	TRUE