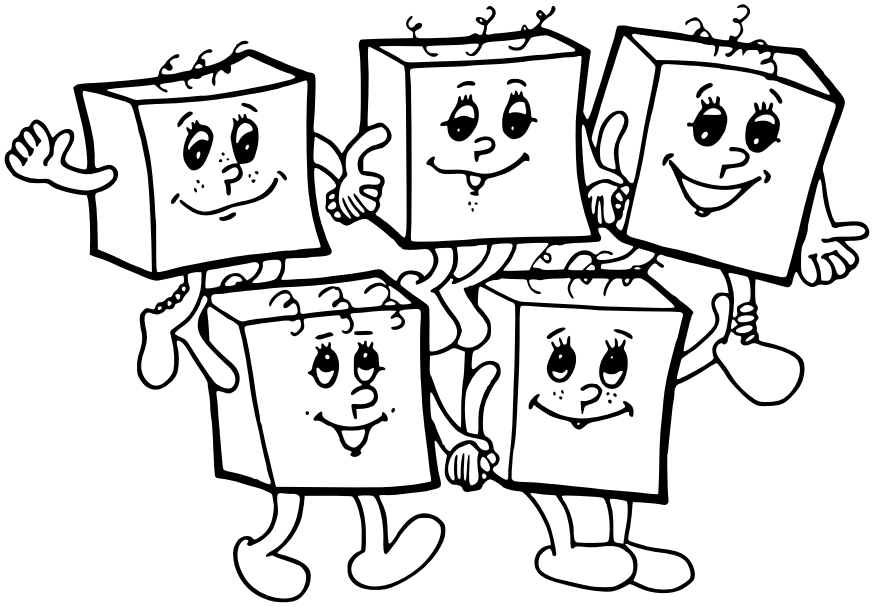
OLYMPIÁDA V INFORMATIKE

NA STREDNÝCH ŠKOLÁCH



dvadsiaty siedmy ročník
školský rok 2011/12

zadania domáceho kola
kategórie A a B

- **Olympiáda v informatike** je od školského roku 2006/07 samostatnou súťažou. Predchádzajúcich 21 ročníkov tejto súťaže prebiehalo pod názvom **Matematická olympiáda, kategória P** (programovanie).
- Oficiálnu **webstránku** súťaže nájdete na `http://oi.sk/`.

Informácie a pravidlá

Pre koho je súťaž určená?

Kategória B má dve kolá: domáce a krajské.
Do kategórie B sa smú zapojiť len tí žiaci základných a stredných škôl, ktorí ešte ani v tomto, ani v nasledujúcom školskom roku nebudú končiť strednú školu.
Kategória A má tri kolá: domáce, krajské a celoštátne.
Do kategórie A sa môžu zapojiť všetci žiaci (základných aj) stredných škôl.
Najlepší riešitelia kategórie A majú šancu reprezentovať Slovensko na medzinárodných súťažiach.

Priebeh súťaže

Za každú úlohu domáceho kola sa dá získať od 0 do 10 bodov. Na základe bodov domáceho kola stanoví Slovenská komisia OI (SK OI) pre každú kategóriu bodovú hranicu potrebnú na postup do **krajského kola**. Očakávame, že táto hranica bude približne rovná **tretine maximálneho počtu bodov**.

V krajskom kole riešitelia riešia štyri teoretické úlohy, ktoré môžu tematicky nadväzovať na úlohy domáceho kola. V kategórii B súťaž týmto kolom končí. V kategórii A prebehne po opravení riešení koordinácia bodovacích škál, spoja sa výsledkové listiny do jednej celoštátnej a podľa nej je približne najlepších 30 riešiteľov pozvaných do **celoštátneho kola**.

V celoštátnom kole účastníci prvý deň riešia tri teoretické úlohy, druhý deň dve praktické úlohy. Najlepší riešitelia sú vyhlásení za víťazov. Približne desať najlepších riešiteľov následne SK OI pozve na týždňové výberové sústredenie. Podľa jeho výsledkov SK OI vyberie družstvá pre Medzinárodnú olympiádu v informatike (IOI) a Stredoeurópsku olympiádu v informatike (CEOI).

Odovzdávanie riešení domáceho kola

Všetky úlohy **je nutné odovzdať prostredníctvom webového rozhrania** na stránkach olympiády (`http://oi.sk/`) najneskôr v deň uvedený v zadaniach príslušnej kategórie.

Ako majú vyzerať riešenia úloh?

V praktických úlohách je vašou úlohou vytvoriť program, ktorý bude riešiť zadanú úlohu. Program musí byť v prvom rade korektný a funkčný, v druhom rade sa snažte aby bol čo najefektívnejší.

V kategórii B môžete použiť ľubovoľný programovací jazyk.

V kategórii A musíte písať riešenia v jazyku Pascal, C, alebo C++, Odovzdaný program bude automaticky otestovaný na viacerých vopred pripravených testovacích vstupoch. Podľa toho, na koľko z nich dá správnu odpoveď, vám budú pridelené body. Výsledok testovania sa dozviete krátko po odovzdaní. Ak váš program nezíska plný počet bodov, budete ho môcť vylepšiť a odovzdať znova, až do uplynutia termínu na odovzdávanie.

Presný popis, ako majú vyzerať riešenia praktických úloh (napr. realizáciu vstupu a výstupu), nájdete na webstránke, kde ich budete odovzdávať.

Ak nie je v zadaní povedané ináč, riešenia teoretických úloh musia v prvom rade obsahovať **podrobný slovný popis použitého algoritmu, zdôvodnenie jeho správnosti** a diskusiu o efektivite zvoleného riešenia (t. j. posúdenie časových a pamäťových nárokov programu). Na záver riešenia uveďte program napísaný v jazyku Pascal, C alebo C++. Ak používate v programe netriviálne algoritmy alebo dátové štruktúry (napr. rôzne súčasti STL v C++), súčasťou popisu algoritmu musí byť dostatočný popis ich implementácie.

Usporiadateľ súťaže

Olympiádu v informatike (OI) vyhlasuje *Ministerstvo školstva SR* v spolupráci so *Slovenskou informatickou spoločnosťou* a *Slovenskou komisiou Olympiády v informatike*. Súťaž organizuje *Slovenská komisia OI* a v jednotlivých krajoch ju riadia *krajské komisie OI*. Na jednotlivých školách ju zaisťujú učitelia informatiky. Celoštátne kolo OI, tlač materiálov a ich distribúciu po organizačnej stránke zabezpečuje IUVENTA v tesnej súčinnosti so Slovenskou komisiou OI.

Zadania kategórie B

Túto kategóriu môžu riešiť len žiaci, ktorí ani v tomto, ani v nasledujúcom školskom roku nematurujú.
Jej riešenia je potrebné odovzdať na stránke `http://oi.sk/` najneskôr **25. 11. 2011**.

B-I-1 Teploty

Janka pripravuje softvér pre letisko, ktorý bude vypisovať štatistiku o teplote vzduchu nad pristávacou dráhou. Pri dráhe má senzor, ktorý pravidelne pošle do jej počítača aktuálnu teplotu. Jej program musí vždy, keď príde nová hodnota, vypísať tri údaje: najmenšiu, priemernú a najväčšiu teplotu za posledných t meraní.

Napíšte program, ktorý bude tieto tri údaje počítať a vypisovať. Pre vás sme už pre jednoduchosť merania teploty pripravili vo forme vstupného súboru.

Formát vstupu

V prvom riadku vstupného súboru sú dve celé čísla n a t: celkový počet meraní a dĺžka intervalu, ktorý nás zaujíma. V každom z nasledujúcich n riadkov je jedno reálne číslo s presne dvomi desatinnými miestami: nameraná teplota v stupňoch Celzia. Tieto záznamy sú uvedené v poradí, v akom boli namerané. Všetky teploty sú z rozsahu od $-80.00\,^{\circ}\mathrm{C}$ po $60.00\,^{\circ}\mathrm{C}$.

(Pre zaujímavosť: Vstupy `3.txt` a `5.txt` obsahujú merania približne zodpovedajúce reálnemu vývoju teplôt niekedy počas roku 2011. Teploty pre tieto vstupy boli merané zhruba raz za 10 sekúnd.)

Formát výstupu

Pre každé meranie začínajúc t-tym (vtedy máme prvýkrát k dispozícii t hodnôt) a končiac posledným n-tým vypíšte jeden riadok a v ňom postupne tri údaje: najmenšie z posledných t meraní, priemer posledných t meraní a najväčšie z posledných t meraní.

Najmenšiu a najväčšiu teplotu vypisujte na aspoň dve desatinné miesta, priemernú na aspoň štyri. Drobné zaokrúhľovacie chyby budeme ignorovať, netrápte sa nimi. V Pascale môžete použiť na výpis priemernej teploty príkaz „`writeln(priemerna_teplota:0:4);`“, v C/C++ príkaz „`printf("%.4f",priemerna_teplota);`“.

Príklad

vstup

```
7 3
-23.15
-23.21
-23.23
-23.24
-23.12
-23.19
-23.20
```

výstup

```
-23.23 -23.1967 -23.15
-23.24 -23.2267 -23.21
-23.24 -23.1967 -23.12
-23.24 -23.1833 -23.12
-23.20 -23.1700 -23.12
```

Prvá priemerná hodnota (-23.1967) je priemerom prvých troch teplôt zo vstupu (-23.15, -23.21 a -23.23). Druhá priemerná hodnota je priemerom druhej, tretej a štvrtej nameranej teploty. A tak ďalej.

Odovzdávanie riešení

Toto je praktická úloha. Napíšte **v ľubovoľnom programovacom jazyku** program, ktorý ju rieši.
Zo stránky `http://oi.sk/` stiahnite ZIP archív obsahujúci 5 testovacích vstupov, nazvaných `1.txt` až `5.txt`.
Vyrobte k čo najviac vstupom správne výstupy a uložte ich do súborov `sol1.txt` až `sol5.txt`.
Odovzdajte ZIP archív obsahujúci **zdrojový kód vášho programu** a tieto výstupné súbory.
Za každý správny výstupný súbor získate 2 body.

B-I-2 Sudoku

Sudoku už dnes pozná snáď každý. Teda okrem Kleofáša, ktorý posledné tri roky prežil v lese s vlkmi. Zadaním sudoku je štvorec 9×9, v ktorého niektorých políčkach sú cifry od 1 po 9. Úlohou riešiteľa je vyplniť všetky zvyšné políčka, a to tak, aby sa v každom riadku, v každom stĺpci aj v každom zvýraznenom štvorci 3×3 každá cifra vyskytovala práve raz. Vpravo vidíte príklad zadania a správneho riešenia.

Kleofáš sa začal učiť, ako riešiť sudoku. Už objavil tri pravidlá. Všetky si ukážeme na zadaní na obrázku vpravo.

- *Prvé pravidlo:* Pozrime sa na štvrtý stĺpec.
  Je v ňom už len jedno voľné políčko a tam musí byť cifra 3.
- *Druhé pravidlo:* Pozrime sa na políčko v poslednom riadku v šiestom stĺpci.
  Jediná cifra, ktorá tam ešte môže byť, je cifra 7.
- *Tretie pravidlo:* Skúsme do siedmeho riadku umiestniť cifru 9.
  Musí byť v piatom stĺpci, nikam inam ju už dať nesmieme.

Každé pravidlo sa samozrejme dá použiť na ľubovoľný riadok, stĺpec, či zvýraznený štvorec 3×3. V treťom pravidle môžeme skúšať umiestniť ľubovoľnú cifru.

Napíšte program, ktorý bude riešiť sudoku. Váš program to samozrejme môže robiť ľubovoľným spôsobom, ale na plný počet bodov stačí, ak bude vedieť správne používať Kleofášove tri pravidlá.

6	2		5	3			1	4
			2		8			
		4	6		1	2		9
8			9		5			7
	4	6	7		2	3	9	
1	9		4		3			5
		5	8		4	1		
			1		6			
9	8			5			4	2

hore: zadanie sudoku
dole: jeho riešenie

6	2	8	5	3	9	7	1	4
7	1	9	2	4	8	5	3	6
3	5	4	6	7	1	2	8	9
8	3	2	9	1	5	4	6	7
5	4	6	7	8	2	3	9	1
1	9	7	4	6	3	8	2	5
2	6	5	8	9	4	1	7	3
4	7	3	1	2	6	9	5	8
9	8	1	3	5	7	6	4	2

Formát vstupu a výstupu

V prvom riadku vstupného súboru je celé číslo t – počet zadaní sudoku, ktoré musíte všetky vyriešiť. Každé zadanie je popísané 9 riadkami. V každom riadku je 9 znakov. Každý znak je cifra od 1 do 9, alebo bodka predstavujúca prázdne políčko. Každé zadanie má jednoznačné riešenie.

Do výstupného súboru vypíšte zaradom riešenia všetkých zadaní.

Na vyriešenie súboru `1.txt` stačí používať prvé pravidlo na riadky a stĺpce.
Na vyriešenie súboru `2.txt` stačí používať prvé pravidlo na riadky, stĺpce a zvýraznené štvorce 3×3.
Na vyriešenie súboru `3.txt` stačí používať prvé dve pravidlá na riadky a stĺpce.
Na vyriešenie súboru `4.txt` stačí používať všetky tri pravidlá na riadky a stĺpce.
Na vyriešenie súboru `5.txt` stačí používať všetky tri pravidlá na riadky, stĺpce a zvýraznené štvorce 3×3.

Príklad

vstup

```
1
9421.83.6
8..94...2
..5.6.4.8
.18479625
5.......3
.6432518.
..1.3.2..
457.9...1
2..5.7964
```

výstup

```
942158376
876943512
135762498
318479625
529681743
764325189
691834257
457296831
283517964
```

Na vyriešenie tohto zadania sudoku stačí používať prvé dve pravidlá na riadky a stĺpce.

Odovzdávanie riešení

Toto je praktická úloha. Napíšte **v ľubovoľnom programovacom jazyku** program, ktorý ju rieši.
Zo stránky `http://oi.sk/` stiahnite ZIP archív obsahujúci 5 testovacích vstupov, nazvaných `1.txt` až `5.txt`.
Vyrobte k čo najviac vstupom správne výstupy a uložte ich do súborov `sol1.txt` až `sol5.txt`.
Odovzdajte ZIP archív obsahujúci **zdrojový kód vášho programu** a tieto výstupné súbory.
Za každý správny výstupný súbor získate 2 body.

B-I-3 Uhádni deň

„Kedy máš narodeniny?“ opýtala sa Anička.
„V máji,“ odpovedal Miško, „ale presný deň musíš uhádnuť!“
„Mal si tento rok narodeniny v utorok?“
„Áno.“
„Je to neskôr ako dvanásteho?“
„Nie.“
„A je to jednociferné číslo?“
„Nie.“
„Tak potom už viem! Desiateho!“ uhádla Anička.

Súťažná úloha

Podúloha a) *(3 body)*

Anička deň Miškovych narodenín uhádla už po tretej otázke. Ale to len preto, že mala šťastie. Ak by mal Miško narodeniny inokedy, mohlo sa jej stať, že by musela položiť otázok oveľa viac.

Nájdite nejakú stratégiu, ako sa pýtať otázky tak, aby ste vedeli *zaručiť*, že hľadaný deň uhádnete po najviac 5 otázkach. Každá otázka musí byť taká, aby na ňu Miško vedel odpovedať „áno“ alebo „nie“.

(Stratégiu stačí popísať slovne, napr.: „Najskôr sa ho opýtam, či mal tento rok narodeniny cez víkend. Ak áno, tak druhá otázka bude ... a ak nie, tak druhá otázka bude ...“)

Podúloha b) *(4 body)*

Aj Anička chce, aby Miško uhádol, kedy má narodeniny. Prezradila mu, že je to v decembri. Jej hra však má trochu iné pravidlá: Miško sa nemôže otázky pýtať postupne. Musí ich najskôr všetky napísať na papier, a až potom mu Anička všetky zodpovie. (Opäť, každá otázka musí mať odpoveď „áno“ alebo „nie“.)

Poraďte Miškovi, aké otázky má napísať na papier a ako potom podľa Aničkiných odpovedí určí deň jej narodenín. Použite čo najmenej otázok.

Podúloha c) *(3 body)*

Nájdite optimálne riešenie predchádzajúcej podúlohy a dokážte, že je optimálne. Inými slovami, zdôvodnite, že keby Miško položil menej otázok, tak nemôže mať istotu, že deň Aničkiných narodenín uhádne.

Odovzdávanie riešení

Toto je teoretická úloha. Pomocou webového rozhrania odovzdajte súbor vo formáte PDF, obsahujúci riešenie, spĺňajúce požiadavky uvedené v pravidlách.

B-I-4 Čierne štvorce

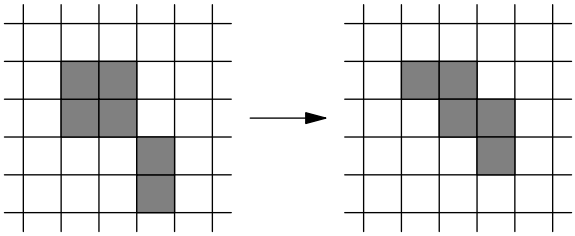
Keď Peťko pozeral priamy prenos z parlamentu, všimol si jedného poslanca. Ten väčšinu času spal. Len keď sa blížilo hlasovanie, zdvihol hlavu, opýtal sa kamaráta naľavo, kamaráta pred sebou a potom podľa toho zahlasoval. A spal ďalej.

Keď sa tak nad tým Peťko zamyslel, napadla mu zákerná otázka: čo by sa stalo, keby tak fungoval nie jeden poslanec, ale úplne všetci? A keďže to je otázka veľmi zložitá, zjednodušil si ju do nasledujúcej podoby:

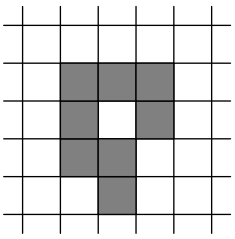
Predstavme si nekonečnú štvorcovú sieť. Na začiatku sú skoro všetky štvorce, ktoré ju tvoria, biele – to sú poslanci, ktorí sú proti aktuálnemu návrhu. Len n štvorcov je čiernych – to sú poslanci, ktorí sú za.

Raz za minútu prebehne všeobecná diskusia. V nej si každý poslanec k svojmu názoru zistí ešte dva ďalšie: názor poslanca naľavo a názor poslanca pred ním. Ten názor, ktorý má medzi týmito troma väčšinu, bude náš poslanec zastávať počas nasledujúcej minúty.

To isté v reči farebných štvorcov: novú farbu štvorca zistíme tak, že sa pozrieme na staré farby jeho, štvorca naľavo od neho a štvorca pod ním. Ako jeho novú farbu vyberieme tú, ktorú vidíme aspoň dvakrát. Tieto zmeny sa udejú naraz pre všetky štvorce. Na obrázku vidíte príklad toho, ako sa po jednej minúte zmení situácia.



Súťažná úloha



Podúloha a) *(2 body)*

Na obrázku je jedna možná začiatočná situácia. Nakreslite, ako bude vyzerať o 1, 2, 3, 4 a 5 minút. (Môžete si buď pomôcť programom, alebo odpovede zostrojiť ručne.)

Podúloha b) *(2 body)*

V príklade v zadaní sa počet čiernych štvorcov zmenšil zo 6 na 5.

Nájdite jedno začiatočné rozmiestnenie čiernych štvorcov, pre ktoré sa po prvej minúte počet čiernych štvorcov nezmenší. (Čiernych štvorcov musí byť konečne veľa a nesmie ich byť nula.)

Podúloha c) *(2 body)*

Nájdite jedno rozmiestnenie 7 čiernych štvorcov také, že ešte aj po 6 minútach bude aspoň jeden štvorec čierny.

Podúloha d) *(4 body)*

Hovoríme, že čierne štvorce *držia pokope*, ak sa z každého na každý dá prejsť tak, že ideme len po čiernych štvorcoch a v každom kroku sa pohneme o políčko jedným z ôsmich základných smerov – ako kráľ v šachu. (Teda stačí, aby štvorce po kope držali rohom, ako na prvom obrázku vľavo.)

Nech n a k sú prirodzené čísla také, že platí $k \leq n$. Pre ktoré dvojice (n, k) sa dá rozmiestniť presne n čiernych štvorcov tak, aby držali pokope a zároveň platilo, že po k minútach budú prvýkrát všetky štvorce biele?

Odovzdávanie riešení

Toto je teoretická úloha. Pomocou webového rozhrania odovzdajte súbor vo formáte PDF, obsahujúci riešenie, spĺňajúce požiadavky uvedené v pravidlách.

Zadania kategórie A

Riešenia kategórie A je potrebné odovzdať na stránke `http://oi.sk/` najneskôr **15. 11. 2011**.

A-I-1 Bezpečná planéta

Planéty vo vesmíre môžeme rozdeliť na tri druhy:
- *typ 0: neobyvateľné* planéty, na ktorých človek okamžite zahynie,
- *typ 1: nehostinné* planéty, na ktorých človek chvíľu prežije, ale nie dlhodobo,
- *typ 2: prívetivé* planéty, na ktorých človek môže spokojne žiť.

Všetky rasy žijúce vo vesmíre cestujú medzi planétami prostredníctvom siete **jednosmerných** teleportov.

Planétu voláme *bezpečná*, ak platí, že je typu 2, a takisto sú typu 2 všetky planéty, na ktoré sa z nej vieme dostať pomocou cestovania teleportami (možno aj viacerými po sebe). Ak teda vysadíte človeka na bezpečnej planéte, máte istotu, že všade, kam odtiaľ docestuje, môže spokojne žiť.

Planétu voláme *znesiteľná*, ak síce nie je bezpečná, ale dá sa z nej pomocou teleportov (možno viacerých po sebe) dostať na bezpečnú planétu bez toho, aby sme museli navštíviť planétu typu 0. Ak teda vysadíte človeka na znesiteľnej planéte a dáte mu vhodné inštrukcie ako cestovať, časom docestuje živý a zdravý na nejakú bezpečnú planétu. Všimnite si, že planéta typu 0 nemôže nikdy byť znesiteľná.

Daný je počet planét n, počet teleportov m, pre každú planétu jej typ a pre každý teleport odkiaľ kam vedie. Zistite, ktoré planéty sú znesiteľné a ktoré bezpečné.

Formát vstupu

Prvý riadok vstupu obsahuje dve celé čísla n a m. Planéty si očíslujeme od 1 po n. Druhý riadok vstupu obsahuje n medzerami oddelených celých čísel z množiny $\{0, 1, 2\}$: postupne pre každé i typ planéty číslo i. Zvyšok vstupu tvorí m riadkov, každý z nich popisuje jeden teleport. Popis teleportu pozostáva z dvoch čísel planét: odkiaľ a kam vedie.

Vo vstupoch, za ktoré bude dokopy 5 bodov, bude $1 \leq n \leq 500$ a $0 \leq m \leq 10\,000$.

V ostatných vstupoch bude $1 \leq n \leq 100\,000$ a $0 \leq m \leq 200\,000$.

Formát výstupu

Vypíšte dva riadky. V prvom z nich reťazec „`bezpecne:"` a za ním postupnosť čísel bezpečných planét. Čísla vypíšte v usporiadanom poradí a pred každým z nich vypíšte práve jednu medzeru. V druhom riadku vypíšte reťazec „`znesitelne:"` a postupnosť čísel znesiteľných planét. Použite rovnaký formát ako v prvom riadku. (Ak je niektorá z postupností prázdna, bezprostredne za dvojbodkou má nasledovať koniec riadku.)

Príklad

vstup

```
7 8
2 1 2 2 2 2 0
1 2
2 3
3 4
4 5
5 3
2 6
6 7
7 6
```

výstup

```
bezpecne: 3 4 5
znesitelne: 1 2
```

Odovzdávanie riešení

Toto je praktická úloha. Pomocou webového rozhrania odovzdajte **funkčný, odladený program**.
Časový limit pre túto úlohu sú 2 sekundy, pamäťový limit je 256 MB.

A-I-2 Naložená loď

Na planéty, ktoré ešte nemajú teleportové terminály, ich treba priviezť medziplanetárnou kozmickou loďou. Keďže ešte nik nevyvinul technológiu na automatické pilotovanie takejto lode, musí ísť s loďou aj posádka. A tá potrebuje jesť. Tvojou úlohou bude presne naplniť celý potravinový nákladný priestor lode balíčkami s jedlom.

Problém je v tom, že balíčky sa vyrábajú na Zemi, zatiaľ čo loď je na obežnej dráhe. Balíčky teda treba dostať zo Zeme na kozmickú loď. To sa robí tak, že sa zoberie niekoľko balíčkov, naložia sa do kapsuly, tú sa vystrelí zo Zeme na obežnú dráhu, a tam si ju už posádka lode odchytí.

Na Zemi majú k dispozícii n typov kapsúl. Kapsula i-teho typu má kapacitu presne a_i balíčkov – nezmestí sa do nej viac a zároveň ich kvôli správnemu vyváženiu a doletu nesmie byť ani menej. Kapsúl každého typu je k dispozícii dostatočne veľa.

Napíš program, ktorý načíta počet typov kapsúl a ich kapacity a vypočíta **najmenší počet** kapsúl potrebný na naloženie kozmickej lode. Súčet kapacít použitých kapsúl musí byť **presne rovný** kapacite k nákladného priestoru lode.

Formát vstupu

Prvý riadok vstupu obsahuje celé číslo n. Druhý riadok obsahuje n celých čísel $a_1, \ldots, a_n$ oddelených medzerami. Tretí riadok obsahuje kapacitu nákladného priestoru lode k.

Môžete predpokladať, že $a_1 < \cdots < a_n$ a že $a_1 = 1$ (teda vždy existuje spôsob, ako presne celú loď naplniť).

- Vo vstupoch za 3 body bude platiť $n \leq 5$, $a_n \leq 2000$ a $k \leq 30$.
- Vo vstupoch za ďalšie 4 body bude platiť $n \leq 30$, $a_n \leq 2000$ a $k \leq 1\,000\,000$.
- Vo vstupoch za posledné 3 body bude platiť $n \leq 30$ a $k \leq 10^{18}$. Spomedzi týchto vstupov jeden bod bude za vstupy s $a_n \leq 100$, druhý za vstupy s $a_n \leq 300$ a tretí za vstupy s $a_n \leq 2000$.

Všimnite si, že na uloženie premennej k potrebujete použiť 64-bitovú premennú: `long long` v C/C++, `int64` v Pascale. Takéto veľké premenné budete tiež potrebovať na počty kapsúl v optimálnom riešení.

Formát výstupu

V prvom riadku vypíšte najmenší počet kapsúl potrebný na presné naloženie našej kozmickej lode.

V druhom riadku vypíšte jedno ľubovoľné riešenie, ktoré použije tento počet kapsúl. Presnejšie, vypíšte n celých čísel $b_1, \ldots, b_n$ oddelených medzerami, pričom b_i je počet použitých kapsúl veľkosti a_i.

Príklady

vstup

```
4
1 10 100 1000
147
```

výstup

```
12
7 4 1 0
```

Použijeme 1 kapsulu s kapacitou 100, 4 s kapacitou 10 a 7 s kapacitou 1. To je dokopy 1+4+7=12 kapsúl.

vstup

```
3
1 8 10
16
```

výstup

```
2
0 2 0
```

Optimálne riešenie je použiť 2 kapsuly s kapacitou 8.

Odovzdávanie riešení

Toto je praktická úloha. Pomocou webového rozhrania odovzdajte **funkčný, odladený program**.
Časový limit pre túto úlohu sú 4 sekundy, pamäťový limit je 256 MB.

A-I-3 Skladník

Ignác bol skladníkom. Pil alkoholické nápoje, používal vulgarizmy a v pracovnej dobe zväčša spal. Niet divu, že sa mu šéf už dlhé roky vyhrážal, že ho nahradí počítačom. Až jedného dňa tá chvíľa naozaj prišla: Šéf kúpil počítač, posadil ho na vrátnicu skladu a Ignáca prepustil. Tým dosiahol presne rovnaký stav ako doteraz, len počítaču už nemusel platiť žiadnu mzdu.

Netrvalo však dlho a šéf si uvedomil, že počítač dokonca môžu aj zapnúť a používať. Potrebujú však na to vhodný softvér.

Podúloha a) *(4 body)* Napíšte čo najefektívnejší program, ktorý bude spravovať inventár skladu. Na vstupe budú prichádzať informácie o tom, koľko kusov ktorého typu tovaru pribudlo alebo ubudlo. Po každom načítaní novej informácie by váš program mal vypísať dva údaje: celkový počet kusov objektov v sklade a počet rôznych typov objektov v sklade.

Podúloha b) *(6 bodov)* Napíšte ešte jeden program, ktorý tiež bude spravovať inventár skladu. Jeho vstup bude vyzerať rovnako ako vstup prvého programu. Po každom načítaní novej informácie by tento program mal vypísať dva údaje: názov typu veci, z ktorého je aktuálne v sklade najviac kusov, a tento počet kusov. (Ak je viac možných typov vecí, vypíšte ten, ktorý je prvý v abecednom poradí.)

Formát vstupu

V každom riadku vstupu je najskôr jedno nenulové celé číslo δ_i a potom jeden reťazec s_i obsahujúci nanajvýš ℓ znakov anglickej abecedy. Význam tohto riadku je, že počet vecí typu s_i v sklade sa zmenil o δ_i.

Pri písaní programov môžete predpokladať, že $\ell \leq 50$, že počet typov vecí v sklade nikdy neprekročí 100 000 a že celkový počet kusov vecí v sklade nikdy neprekročí 10^{18} (a teda sa zmestí do bežnej celočíselnej premennej). Tiež môžete predpokladať, že nikdy nedostanete inštrukciu, ktorá by počet vecí nejakého typu zmenila na záporný.

Príklad

vstup

```
+3 koberec
+2 buldozer
+1 buldozer
+7 zeriav
-3 buldozer
-5 zeriav
```

výstup, podúloha a)

```
kusov: 3, typov: 1
kusov: 5, typov: 2
kusov: 6, typov: 2
kusov: 13, typov: 3
kusov: 10, typov: 2
kusov: 5, typov: 2
```

výstup, podúloha b)

```
koberec 3
koberec 3
buldozer 3
zeriav 7
zeriav 7
koberec 3
```

Hodnotenie

Každá podúloha sa hodnotí samostatne.

Ako súčasť svojho riešenia odhadnite, ako **najdlhšie** (t.j. v najhoršom možnom prípade) môže každému z vašich programov trvať spracovanie n-tej inštrukcie. (Pri odhade môžete použiť premennú t pre aktuálny počet typov vecí v sklade, premennú k pre aktuálny počet kusov vecí v sklade a premennú ℓ pre maximálnu dĺžku názvu veci.) Táto časová zložitosť bude hlavným kritériom hodnotenia. Snažte sa teda, aby váš program každú informáciu zaručene spracoval rýchlo.

Pripomíname: Ak používate v programe netriviálne algoritmy alebo dátové štruktúry (napr. rôzne súčasti STL v C++), súčasťou popisu algoritmu musí byť dostatočný popis ich implementácie, prípadne popis implementácie príbuznej dátovej štruktúry s rovnakou časovou zložitosťou operácií.

Odovzdávanie riešení

Toto je teoretická úloha. Pomocou webového rozhrania odovzdajte súbor vo formáte PDF, obsahujúci riešenie, spĺňajúce požiadavky uvedené v pravidlách.

A-I-4 Zlomkové programy

V tomto ročníku olympiády sa budeme v každom súťažnom kole stretávať so zlomkovými programmi. V študijnom texte uvedenom za zadaním tejto úlohy je popísané, ako tieto programy fungujú.

Súťažná úloha

a) (5 bodov) Na vstupe je číslo n tvaru $2^x 3^y 5$, pričom $x, y \geq 0$.
Napíšte program, ktorý ho prerobí na číslo 5, ak $x = y$, resp. na číslo 7, ak $x \neq y$.

b) (5 bodov) Na vstupe je číslo n tvaru $2^x 3$, pričom $x \geq 0$.
Napíšte program, ktorý ho prerobí na číslo tvaru 2^y, kde $y = \lfloor x/2 \rfloor$.

Odovzdávanie riešení

Toto je teoretická úloha. Pomocou webového rozhrania odovzdajte súbor vo formáte PDF, obsahujúci riešenie, spĺňajúce požiadavky uvedené v pravidlách.

Interpreter

Pred tým, než svoje riešenie odovzdáte, si ho môžete otestovať. Na adrese `http://oi.sk/zlomky/` nájdete interpreter zlomkových programov.

Študijný text

Zlomkové programy predstavujú jeden veľmi jednoduchý spôsob, ako počítať niektoré funkcie na prirodzených číslach. Samotný zlomkový program je veľmi jednoduchý: je to konečná postupnosť zlomkov, teda kladných racionálnych čísel $(z_1, \ldots, z_k)$.

Výpočet zlomkového programu prebieha v krokoch. Počas výpočtu si udržiavame jediné celé číslo, tzv. *aktuálnu hodnotu* a. Na začiatku výpočtu na vstupe n je $a = n$. Každý krok výpočtu vyzerá nasledovne: Nájdeme najmenšie i také, že $a \cdot z_i$ je celé číslo, a zmeníme aktuálnu hodnotu na $a \cdot z_i$. Ak také i neexistuje, výpočet končí.

Príklad 1. Ukážeme si program, ktorý pre vstup $n = 2^x$ (kde $x \geq 0$) vyrobí výstup 3^y, kde $y = x \bmod 3$.

Jedným takýmto programom je postupnosť $(1/8, 9/4, 3/2)$. Slovne si priebeh výpočtu tohto programu môžeme popísať nasledovne: kým sa to dá, zmenšuj x o 3. Keď sa to už nedá, máme v a číslo 1, 2, alebo 4, vyrobíme z neho teda 1, 3, alebo 9.

Príklad výpočtu pre $n = 1024 = 2^{10}$: Hodnota a sa bude meniť nasledovne: $2^{10} \xrightarrow{1} 2^7 \xrightarrow{1} 2^4 \xrightarrow{1} 2 \xrightarrow{3} 3$.
(Číslo nad šípkou je poradovým číslom zlomku, ktorým sme a v danom kroku prenásobili.)

Všimnite si, že záleží na poradí zlomkov. Napríklad program $(9/4, 3/2, 1/8)$ by z čísla 2^x vyrobil číslo 3^x. Zlomok $1/8$ by sa pri výpočtoch tohto programu nikdy nepoužil.

Iné vyhovujúce programy sú $(1/8, 3/2)$, $(3/2, 1/27)$ a $(1/27, 3/2)$. Rozmyslite si, prečo každý z nich tiež rieši zadanú úlohu.

Príklad 2. Čo urobí program $(2/2)$ na vstupe $n = 4$? A čo na vstupe $n = 7$?

Obľúbenou chybou je odpovedať, že na vstupe $n = 4$ sa tento program zacyklí, ale na vstupe $n = 7$ sa program zastaví, lebo 7 nie je deliteľné dvomi. Správna odpoveď ale je, že v **oboch** prípadoch bude program bežať do nekonečna. Zaujímajú nás totiž len hodnoty zlomkov, nie ich zápis. Zlomok $2/2$ predstavuje racionálne číslo 1, a $7 \cdot (2/2) = 7 \cdot 1$ je celé číslo.

Aby ste sa vo svojich riešeniach takto nepoplietli, odporúčame uvádzať všetky zlomky v základnom tvare. Pre takto zapísanú postupnosť zlomkov potom už platí, že v každom kroku hľadáme najmenšie i, pre ktoré menovateľ i-teho zlomku delí aktuálnu hodnotu.

Príklad 3. Ukážeme si program, ktorý pre vstup $n = 2^{x+1}$ (kde $x \geq 0$) vyrobí výstup 3^{x+2}.

Číslo na vstupe je určite párne a nie je deliteľné žiadnym prvočíslom iným ako 2. Použijeme prvočíslo 11 ako značku, že už nie sme na začiatku výpočtu. V prvom kroku teda prepíšeme číslo 2^{x+1} na $2^x 3^2 11$. Toto dosiahneme zlomkom $3^2 \cdot 11/2 = 99/2$.

Ak už vidíme v prvočíselnom rozklade aktuálnej hodnoty prvočíslo 11, znamená to, že prvý krok máme úspešne za sebou. Môžeme teda spokojne „meniť dvojky na trojky“. To vieme dosiahnuť napríklad postupnosťou zlomkov $(3 \cdot 13)/(2 \cdot 11)$ a $11/13$. (Všimnite si, že nestačí použiť jeden zlomok $33/22$. Ak vám nie je jasné, prečo, prečítajte si ešte raz príklad 2.)

Časom sa takto dostaneme k číslu $3^{x+2}11$. V tejto situácii už stačí len vydeliť aktuálnu hodnotu číslom 11 a môžeme skončiť.

Pozor treba dať na to, že vyššie popísané zlomky treba dať do správneho poradia: $(39/22, 11/13, 1/11, 99/2)$. V prvom kroku výpočtu sa zjavne použije posledný zlomok. Od tejto chvíle je aktuálna hodnota deliteľná číslom 11 alebo 13. Striedavo sa používajú prvé dva zlomky, až kým sa nedostaneme do situácie $a = 3^{x+2}11$. Vtedy sa už prvý ani druhý zlomok použiť nedá. Použije sa preto tretí, čím dosiahneme $a = 3^{x+2}$ a výpočet zjavne končí.

Poznámka. V riešeniach podobných tomu v príklade 3 nie je nutné čitatele a menovatele zlomkov roznásobovať. Pokojne uveďte svoje riešenie v tvare $\big((3 \cdot 13)/(2 \cdot 11),\ 11/13,\ 1/11,\ (3^2 \cdot 11)/2\big)$.

SLOVENSKÁ KOMISIA OLYMPIÁDY V INFORMATIKE

DVADSIATY SIEDMY ROČNÍK OLYMPIÁDY V INFORMATIKE

Vydala IUVENTA s finančnou podporou Ministerstva školstva SR
Náklad: 400 výtlačkov
Zodpovedný redaktor: Michal Forišek
Sadzba programom LaTeX