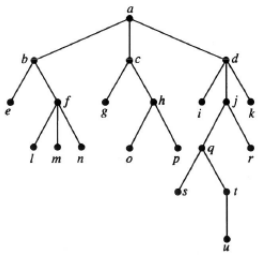
Návodné úlohy k domácemu kolu OI: kategória A

Toto sú návodné úlohy k domácemu kolu 34. ročníka Olympiády v informatike. Ide teda o sadu ľahších úloh, ktoré tematicky súvisia so súťažnými úlohami. Riešenie týchto úloh môže byť dobrou prípravou na riešenie súťažných úloh. Za riešenia týchto návodných úloh nie sú žiadne body do súťaže.

A-I-1 Rušenie ulíc

1. Aké vlastnosti musí mať obrázok z čiar, aby platilo, že sa dá celý nakresliť jedným ťahom (bez toho, aby sme zdvihli pero z papiera alebo šli po tej istej čiare dvakrát)?

2. Daný je zakorenený strom (príklad viď obrázok nižšie). Na každý jeho vrchol môžeme (ale nemusíme) položiť mincu.



   Podstrom určený vrcholom x tvoria všetky vrcholy (vrátane x), pre ktoré platí, že cesta z koreňa doň vedie cez x.

   V každom vrchole máme predpísané, či má ním určený podstrom dokopy obsahovať párny alebo nepárny počet mincí.

   Dokážte, že bez ohľadu na to, ako strom vyzerá a aké parity sú predpísané, vždy vieme umiestniť mince tak, aby sme všetky predpísané parity splnili.

   Nájdite efektívny algoritmus, ktorý jedno vhodné rozmiestnenie mincí zostrojí.

3. Ako by sa zmenilo riešenie predchádzajúcej úlohy, keby sme namiesto do vrcholov kládli mince na hrany stromu?

   (Aj v tejto verzii úlohy má každý vrchol predpísanú paritu počtu mincí v jeho podstrome.)

4. Napíšte samostatný program, ktorý efektívne rieši súťažnú úlohu A-I-1 pre $k = 1$, teda pre vstupy, v ktorých je presne jedna ulica, ktorú nesmieme zrušiť.

   Všimnite si, že keď do tohto programu ešte dorobíte triviálne prípady $k = 0$ a $k = m$, dostanete zaň 5 bodov.

A-I-2 Cyklistický závod

1. Napíšte program, ktorý zistí, či daný neorientovaný graf obsahuje cyklus.

2. Upravte predchádzajúci program tak, aby v prípade, že graf cyklus obsahuje, jeden cyklus aj našiel a vypísal.

3. Daný je acyklický graf a v ňom sú vyznačené dva vrcholy A a B. Napíšte program, ktorý zistí, či majú všetky možné cesty z A do B rovnakú dĺžku.

A-I-3 Trojuholník

1. Ako vypočítať obsah trojuholníka, ak poznáme dĺžky jeho strán?

2. Ako vypočítať obsah trojuholníka, ak poznáme súradnice jeho vrcholov?

   Jedno možné riešenie je samozrejme vypočítať si z nich dĺžky strán. Existujú však aj šikovnejšie spôsoby.

   Ak sú všetky súradnice vrcholov celé čísla, je dvojnásobok obsahu trojuholníka nutne tiež celé číslo. Toto celé číslo vieme vypočítať zo súradníc vrcholov len pomocou celočíselnej aritmetiky. Ako?

3. Daných je n rôznych bodov v rovine. Dvojica daných bodov sa vidí, ak na úsečke, ktorá ich spája, neleží žiaden iný z daných bodov.

   Napíšte program, ktorý zistí, či sa všetky dvojice bodov vidia. Ak nie, program by mal nájsť jednu konkrétnu dvojicu bodov, ktoré sa nevidia.

A-I-4 Online algoritmy

Miško túto zimu pôjde niekoľkokrát lyžovať. Presnejšie, lyžovať pôjde vždy, keď ho jeho priateľka vytiahne na lyže. Miško vôbec netuší, koľkokrát sa to stane.

Tiež je dôležitá skutočnosť, že Miško nemá vlastné lyže. Vždy, keď má ísť lyžovať, stojí pred dilemou: má si lyže na ten jeden deň požičať, či si už má kúpiť vlastné?

Kúpiť si lyže je presne n-krát drahšie ako požičať si ich (pričom $n > 1$ je nejaké konkrétne prirodzené číslo). Miško by samozrejme bol rád, keby za celú zimu dokopy minul čo najmenej peňazí.

1. Rozcvička 1: Dokážte, že algoritmus „rovno si kúpim lyže“ je n-kompetitívny.

   (Teda dokážte, že bez ohľadu na to, ako bude nadchádzajúca zima vyzerať, zaplatí Miško dokopy nanajvýš n-krát toľko ako by zaplatil, keby vopred poznal celú budúcnosť.)

2. Rozcvička 2: Dokážte, že algoritmus „vždy si budem požičiavať“ nie je c-kompetitívny pre žiadnu kladnú konštantu c.

3. Uvažujme teraz nasledujúci algoritmus: „Prvých n-krát si lyže požičiam a keď budeme mať ísť $(n+1)$. raz lyžovať tak si ich kúpim.“

   Koľko-kompetitívny je tento algoritmus a prečo?

4. Je algoritmus z predchádzajúcej úlohy optimálnym online algoritmom pre túto úlohu?

Návodné úlohy k domácemu kolu OI: kategória B

Toto sú návodné úlohy k domácemu kolu 34. ročníka Olympiády v informatike. Ide teda o sadu ľahších úloh, ktoré tematicky súvisia so súťažnými úlohami. Riešenie týchto úloh môže byť dobrou prípravou na riešenie súťažných úloh. Za riešenia týchto návodných úloh nie sú žiadne body do súťaže.

B-I-1 Kopa tričiek

1. Skupina n priateľov, očíslovaných od 1 po n, išla do karaoke baru, kde teraz postupne po jednom spievajú. Niektorí z nich už boli zaspievať po jednej pesničke, ostatní ešte neboli spievať vôbec.

   Samozrejme, vždy by mal ako ďalší ísť spievať niekto, kto zatiaľ spieval najmenejkrát.

   Na vstupe sú čísla ľudí, ktorí už raz spievali. Napíšte program, ktorý ich prečíta a následne bude bežať do nekonečna a vypisovať, kto má ísť ako ďalší v poradí spievať.

2. Jano má podobnú kopu tričiek ako mal Žaba. Jano ju však používa ináč. Každé ráno si zoberie tričko z kopy a každý večer hodí použité tričko do koša na prádlo. Raz za čas potom všetky tričká z koša vyperie a vráti (v ľubovoľnom poradí) späť na kopu.

   Napíšte program, ktorý bude čo najefektívnejšie simulovať Jana. Program teda na začiatku načíta zoznam tričiek v kope a potom bude spracúvať požiadavky tvarov „zober tričko z vrchu kopy a oznám, ktoré to je“ a „vráť všetky opraté tričká späť na kopu“.

3. Peťka funguje rovnako ako Jano, ale keď vracia tričká na kopu, vráti ich vždy do ich pôvodného poradia. Napíšte program, ktorý bude simulovať Peťku.

   Dajte si pozor, aby váš program na vrátenie opratých tričiek na kopu potreboval len čas priamo úmerný počtu *čerstvo opratých* tričiek (a nie počtu *úplne všetkých* tričiek).

B-I-2 O Jankinom bratovi

1. Napíšte program, ktorý bude hrať tú istú hru ako Miško, ale bude to robiť *pažravo*: vždy, keď môže nejaké slovo zobrať, tak ho zoberie. Inými slovami, vždy, keď to slovo, ktoré ste práve dostali, nadväzuje na slovo, ktoré ste naposledy zobrali, tak ho musíte zobrať. Začnete vždy tým, že zoberiete úplne prvé slovo, ktoré dostanete.

2. Program z predchádzajúcej úlohy Miškovu hru nehrá optimálne – existujú vstupy, pre ktoré Miško nájde lepšie riešenie ako váš program. Nájdite jeden takýto vstup. Koľko najmenej slov môže mať?

3. Napíšte program, ktorý načíta reťazec a vypočíta, koľkými spôsobmi sa dá v danom reťazci vyznačiť tri pozície tak, aby na najľavejšej bolo písmeno P, na prostrednej E a na najpravejšej S.

   (Inými slovami, program má spočítať, koľko je v danom reťazci výskytov podreťazca PES.)

4. Viete predchádzajúcu úlohu vyriešiť tak, aby hľadaný program mal lineárnu časovú zložitosť? (Teda počet krokov, ktoré spraví, by mal byť približne priamo úmerný dĺžke reťazca.)

5. A viete to dokonca spraviť tak, aby mal váš program nie len lineárnu časovú, ale navyše aj konštantnú pamäťovú zložitosť? Takýto program si teda ani len nesmie naraz zapamätať celý vstupný reťazec. Môže ho len raz postupne znak po znaku prečítať a počas toho si priebežne o ňom počítať a v bežných celočíselných premenných pamätať nejaké informácie.

B-I-3 Cudzokrajná abeceda

1. Prišli dvaja cudzokrajní hostia. Dostal si ich vizitky a z tých si zistil ich mená: reťazce X a Y. Hostí chceš predstaviť v abecednom poradí. Podobne ako v súťažnej úlohe máš ale problém: nevieš, aké poradie písmen má ich abeceda.

   Na svojom smartfóne si stíhaš pre *jednu jedinú dvojicu písmen* zistiť, ktoré z nich je skôr v cudzineckej abecede. Ale ktorú dvojicu máš dať hľadať?

   Napíšte program, ktorý načíta reťazce X a Y a vypíše, ktorú dvojicu písmen si treba vyhľadať.

   Napr. pre mená `Jozo` a `Fero` by mal program vypísať „zisti si, či je v cudzej abecede skôr `J` alebo `F`.“

2. Ak si mal v minulej úlohe ozaj šťastie, tak sa ti mohlo stať, že si nepotreboval na telefóne hľadať vôbec nič. Kedy presne taká situácia nastáva?

3. Inštalovanie rôznych druhov softvéru často funguje nasledovným spôsobom: Softvér je rozdelený do rôznych samostatných *balíčkov*. Každý balíček B má sadu *závislostí* – teda zoznam iných balíčkov, ktoré treba nainštalovať *skôr ako samotný balíček* B. Až keď sú všetky závislosti splnené, dá sa nainštalovať samotný balíček B.

   Napríklad balíček obsahujúci prostredie na tvorbu hier potrebuje balíček na prácu so zvukom a tri rôzne balíčky na prácu s rôznymi grafickými formátmi. Jeden z tých grafických balíčkov ďalej potrebuje balíček s knižnicou pre kompresiu a dekompresiu.

   Inštalovanie softvéru potom funguje tak, že špeciálnemu programu (tzv. správca balíčkov) poviete, ktorý balíček si chcete nainštalovať. Správca balíčkov následne zistí a v správnom poradí nainštaluje všetky potrebné balíčky.

   Akým spôsobom to celé funguje? Skúste navrhnúť postup, ktorým správca balíčkov zabezpečí, že nájde a v správnom poradí nainštaluje všetky potrebné balíčky.

4. V predchádzajúcej úlohe by mohol nastať problém: Čo keby napríklad niekto naschvál vytvoril dva balíčky A a B také, že A hovorí „najskôr musíš nainštalovať B“ a naopak B hovorí „najskôr musíš nainštalovať A“? V takejto situácii sa samozrejme nedá nainštalovať ani jeden z nich.

   Zamyslite sa nad vaším riešením predchádzajúcej úlohy. Čo by váš program spravil, keby ste mu dali takéto dva zákerné balíčky a požiadali by ste ho, nech nainštaluje A?

   Vo všeobecnosti takéto problémy nastanú vždy, keď sa v závislostiach medzi balíčkami nachádza nejaký *cyklus*. Skúste navrhnúť, ako upraviť program pre správcu balíčkov tak, aby keď stretne cyklus v závislostiach, korektne skončil a informoval používateľa o nájdenom probléme.

B-I-4 Šimon a UFO

1. Na číselnej osi na súradnici 0 stojí robot. Robot vie spraviť krok o 1 doprava aj krok o 1 doľava.

   Na neznámej celočíselnej súradnici x rastie jahoda. Keď robot príde na túto súradnicu, jahodu nájde.

   Ako máme robota naprogramovať, ak chceme, aby jahodu zaručene našiel, bez ohľadu na hodnotu x?

2. Na nekonečnej štvorcovej sieti na políčku $(0, 0)$ stojí druhý robot. Tento robot sa vie hýbať po políčkach. Presnejšie, vieme mu dať inštrukcie, aby sa pohol o jedno políčko smerom na sever / juh / východ / západ.

   Na niektorom inom políčku (x, y) rastie malina.

   Naprogramujte aj tohto robota tak, aby zaručene po konečnom počte krokov malinu našiel.

3. Rádovo koľko krokov (ako funkciu čísla x) spravil váš robot z prvej úlohy?

   A nejde to aj lepšie?