Pravidlá

Riešenia tejto písomnej skúšky je potrebné odovzdať do **soboty 29. 12. 2018, 10:00** e-mailom na `misof@ksp.sk`. Ak odovzdávate viacero súborov, zozipujte ich prosím. Počas riešenia priebežne **sledujte oznamy** na webstránke predmetu.

Smiete využívať ľubovoľné neživé zdroje informácií ktoré existovali v okamihu začiatku písomky. Pochopiteľne, vrátane všetkého zverejneného na stránke predmetu. Až do deadline je zakázané akýmkoľvek spôsobom diskutovať o úlohách s kýmkoľvek živým. Použité externé zdroje dostatočne **adekvátne citujte**. Čo viete odcitovať, netreba rozpisovať. Ak ste pri riešení úlohy písali program, odovzdajte aj ten.

Do hodnotenia sa vám započíta **5 najlepšie vyriešených** úloh. Zapísaniu známky môže predchádzať rozhovor o niektorých úlohách ktoré ste riešili.

0 Nultá podúloha

Do textu riešenia nezabudnite prosím vyplniť dva údaje: celé meno a **odhadovaný čistý čas** (v hodinách) strávený riešením úloh. Tento údaj **nebude mať vplyv na hodnotenie**, chcem ho kvôli lepšej kalibrácii úloh v budúcnosti.

1 Prechádzka na sedem

Daný je neorientovaný graf G. Navrhnite čo najefektívnejší algoritmus, ktorý spočíta, koľkými spôsobmi vieme v grafe G spraviť sedem krokov po hranách tak, aby sme sa nakoniec vrátili tam, kde sme začínali.
(Formálne, chceme počet osmíc vrcholov $(v_0, v_1, \ldots, v_7)$ takých, že $v_0 = v_7$ a $\forall i : v_i v_{i+1}$ je hranou v G. Podotýkame, že v_i nemusia byť navzájom rôzne.)

2 Lampy v Kocúrkove

Kocúrkovo má stromovú topológiu. Vrcholy stromu voláme lokality a ich počet značíme n. Hrany voláme ulice. Každá ulica má pouličné osvetlenie, ktoré je buď zapnuté alebo vypnuté.
Na začiatku dostanete popis topológie Kocúrkova. Potom budete dostávať postupnosť požiadaviek. Každá z nich má buď tvar „rozsvieť všetky ulice na ceste z a_i do b_i“ alebo tvar „zhasni všetky ulice na ceste z a_i do b_i“. Po každej požiadavke treba oznámiť, koľko ulíc v celom Kocúrkove v danej chvíli svieti.
Navrhnite algoritmus, ktorý každú požiadavku spracuje v čase $O(\log^2 n)$.

Partial credit: Ak vyššie uvedenú úlohu neviete riešiť, popíšte aspoň, ako ju riešiť v čase $O(\log n)$ ak všetkých n Kocúrkovských lokalít leží na jednej ceste.

3 Húsky, húsky, poďte domov!

Na vodorovnej rovine je nakreslená nekonečná štvorcová sieť a na tej je zvolená súradnicová sústava. Na políčku $(0, 0)$ je domček. Na niektorých políčkach siete stoja húsky. Na každé políčko sa zmestí ľubovoľne veľa húsok.
S húskami hrajú Augustín a Marián nasledujúcu zábavnú hru. Hráči ťahajú striedavo, začína Augustín. Hráč na ťahu si musí vybrať ľubovoľnú jednu húsku a tú presunúť z jej aktuálneho políčka na ľubovoľné iné, ktoré je bližšie k domčeku. (Formálne, Euklidovská vzdialenosť stredu nového políčka od stredu domčeka musí byť menšia ako tá istá metrika pre staré políčko. Jedným z možných ťahov je zobrať ľubovoľnú húsku, ktorá nie je v domčeku, a presunúť ju do domčeka. Húskou, ktorá je v domčeku, sa už nedá hýbať.)
Prehráva hráč, ktorý už nevie spraviť platný ťah.

a) Na začiatku hry stojí práve jedna húska na každom políčku (x, y) takom, že $|x|$ aj $|y|$ sú prvočísla menšie ako 1000. Ak obaja hráči budú hrať optimálne, kto vyhrá? A akej stratégie sa má držať, aby vyhral?

b) Na začiatku hry stojí práve jedna húska na každom z 167 políčok so súradnicami (p_i, p_{i+1}), kde $p_1 = 2$, $p_2 = 3$, $\ldots$, $p_{168} = 997$ sú postupne všetky prvočísla menšie ako 1000. Ak obaja hráči budú hrať optimálne, kto vyhrá a prečo?

(Neostýchajte sa, obzvlášť v časti b, použiť pri riešení úlohy program.)

4 Nejdeme, bojíme sa!

Analyzujte misère verziu hry z predchádzajúcej úlohy – teda hru s rovnakými pravidlami, v ktorej ale **vyhráva** hráč, ktorý už nevie spraviť platný ťah. (A teda prehráva hráč, ktorý dovedie do domčeka poslednú húsku.) Inými slovami, vysvetlite, ako efektívne zistiť, či je daná pozícia tejto hry vyhrávajúca alebo prehrávajúca.
(Efektívne riešenie by si malo poradiť s ľubovoľnou pozíciou v ktorej súradnice húsok v absolútnej hodnote neprekračujú 1000.)

5 Slovenský pásik

Máme pásik papiera rozmerov $3 \times n$. Každý jeho jednotkový štvorec chceme zafarbiť na bielo, modro alebo červeno, a to tak, aby nikde neboli vodorovne alebo zvisle tri susediace políčka rovnakej farby. Koľko platných ofarbení existuje (ako funkcia čísla n, stačí asymptotický odhad)?

6 Hady a rebríky

Hra „Hady a rebríky" sa hrá na hracom pláne, ktorý tvoria políčka očíslované od 0 (štart) do 100 (cieľ). Na pláne je nakreslených niekoľko hadov a rebríkov, ich rola sa líši len kozmeticky. Každý objekt (had alebo rebrík) má svoj začiatok na nejakom políčku z_i a koniec na nejakom inom políčku k_i. Všetky tieto políčka sú navzájom rôzne.
Hra pre jedného hráča pozostáva z kôl. V každom kole hráč hodí 6-stennou kockou a pohne figúrku o toľko políčok, aké číslo hodil. Potom sa pozrie, či na novom políčku začína objekt, a ak áno, posunie figúrku po objekte na jeho koniec. (Do cieľa netreba prísť presne – napr. ak hráč stojí na políčku 98 a hodí 5, tiež dosiahol cieľ.)
Popíšte, ako by ste k danému hraciemu plánu čo najpresnejšie zistili očakávaný počet kôl hry na ňom. (Môžete predpokladať, že hady a rebríky sú rozmiestnené tak, aby hru išlo v konečnom čase dohrať.)

7 Kódovanie

Prefix-free kód pre abecedu Σ je zobrazenie $k : \Sigma \to \{0,1\}^*$ také, že pre každé $x, y \in \Sigma$, $x \neq y$ platí, že $k(x)$ nie je prefixom $k(y)$. Prefix-free kódy máme radi, lebo ich vieme jednoznačne a efektívne dekódovať.
Máme abecedu Σ a pre každé písmeno z nej je známa jeho relatívna početnosť $f(x)$ v texte, ktorý budeme kódovať. (Súčet všetkých $f(x)$ je 1.) Prefix-free kód k teda v priemere na zakódovanie jedného znaku textu použije $\sum_x f(x) \cdot \, |k(x)|$ bitov. Čím je táto hodnota menšia, tým je kód lepší.
Je známe, že najlepším riešením vyššie popísaného problému je tzv. Huffmanov kód k_h. Ten má ale jednu škaredú vlastnosť: nezachováva lexikografické poradie. Formálne, môže sa stať, že v abecede Σ platí $a < b$, ale reťazec $k_h(a)$ je v lexikografickom poradí väčší od reťazca $k_h(b)$.

Hovoríme, že prefix-free kód k *zachováva poradie* ak pre abecedu Σ ktorú tvoria (postupne v tomto poradí) znaky $s_1 < s_2 < \cdots < s_n$ platí $k(s_1) < k(s_2) < \cdots < k(s_n)$. Napríklad kód, ktorý zakóduje a ako 00, b ako 01, c ako 10, d ako 1100 a e ako 11011 je prefix-free kód ktorý zachováva poradie, lebo $00 < 01 < 10 < 1100 < 11011$.

Nájdite čo najefektívnejší algoritmus, ktorý k daným hodnotám $f(s_i)$ nájde najlepší prefix-free kód zachovávajúci poradie. Za čokoľvek polynomiálne bude veľa bodov.