Domáca úloha č. 4

1-AIN-105, Zima 2023

Termín: 27.11.2023, 22:00

Skôr ako sa pustíte do riešenia domácej úlohy, oboznámte sa so všeobecnými pokynmi, ktoré sú priložené na konci tohto dokumentu. Riešenia, ktoré odovzdáte, musia byť vaše vlastné. Neopisujte a nesnažte sa nájsť riešenia v literatúre alebo na internete!

1. [*20 bodov*] Pre každú z nižšie uvedených rekurencií odvoďte čo najtesnejší asymptotický horný odhad pre $T(n)$. Predpokladajte, že $T(1) = \Theta(1)$ and $T(0) = \Theta(1)$. V prípade, že sa rozhodnete aplikovať master theorem, môžete zanedbať dolné a horné celé časti. Zdôvodnite svoje odpovede.

a) $T(n) = 7T(\lfloor n/2 \rfloor) + n^2$
b) $T(n) = 16T(\lfloor n/4 \rfloor) + n^2$
c) $T(n) = 4T(\lfloor n/2 \rfloor) + n^2\sqrt{n}$
d) $T(n) = T(\lfloor \sqrt{n} \rfloor) + 1$

2. [*20 bodov*] **Plátanie hadíc II.** Na nemenovanej fakulte požiarna inšpekcia našla deravú požiarnu hadicu. Keďže peňazí je málo, údržba sa rozhodla hadicu zaplátať. Hadica má k malých dier vo vzdialenostiach $d_1 < d_2 < \ldots < d_k$ centimetrov od ľavého konca (hadicu si predstavte ako úsečku idúcu zľava doprava). Máme k dispozícii dva typy záplat, prvý typ pokryje 3cm dĺžky, druhý typ pokryje 5cm dĺžky hadice (záplata pokryje aj diery, ktoré sú presne na jej ľavom alebo pravom okraji).

Navrhnite algoritmus, ktorý zistí, ako zalepiť všetky diery s použitím najmenšej možnej celkovej dĺžky záplat.

Vzorový príklad vstupu a riešenia:

$k = 5$, $(d_1, \ldots, d_k) = (1, 3, 4, 6, 10)$

Diery možno pokryť dvoma záplatami s celkovou dĺžkou 8cm. Prvú záplatu o dĺžke 3cm priložíme ľavým okrajom na pozíciu 1cm a pokryje diery d_1, d_2 a d_3, druhú záplatu o dĺžke 5cm priložíme ľavým okrajom na pozíciu 5.5cm a pokryje diery d_4 a d_5.

3. [*20 bodov*] **Najdlhšia prechádzka** (programátorská úloha).

Odovzdávanie úlohy: `https://testovac.ksp.sk/tasks/eads2023-du4a/`

Tí z Vás, ktorí poznajú Emmu vedia, že naozaj *nemá* rada 2 veci: ranné vstávanie a prechádzky. Na druhej strane, tí z Vás, ktorí poznajú Marcela vedia, že naozaj *má* rád 2 veci: ranné vstávanie a prechádzky. Ranné vstávanie nie je problém, ale prechádzanie problém bol. Preto sa dohodli na kompromise: ak sa budú prechádzať, tak iba dole kopcom. Marcel by sa chcel ísť s Emmou prejsť a chcel by teda naplánovať čo najdhlšiu trasu, ktorú je Emma ochotná prejsť. Teraz ale bezradne stojí nad mapou a premýšľa: aká dlhá je taká trasa?

Dostanete obdĺžnikovú mapu rozmerov $r \times c$, kde o každom políčku viete jeho výšku $v_{i,j}$. Vašou úlohou je povedať dĺžku najdlhšej trasy, ktorou môže Marcel s Emmou ísť. Táto trasa musí spĺňať podmienku z kompromisu, teda musí celý čas klesať. To znamená, že ak trasa ide z políčka (a, b) na políčko (c, d), tak musí platiť, že $v_{a,b} > v_{c,d}$.

Formát vstupu: V prvom riadku sú dve medzerou oddelené čísla r, c — počet riadkov a stĺpcov mapy. Nasleduje r riadkov a na každom z nich je medzerou oddelených c čísel — výšky políčok na mape.

Formát výstupu: Na výstupe vypíšte jediné číslo — najdlhšiu trasu akú môže Marcel naplánovať.

Príklady:

vstup:	**výstup:**

`2 2`
`5 4`
`2 3`
	`3`

Najdlhšia cesta začína vľavo hore, ide doprava, dole a doľava. Takto prejde cez všetky políčka.

vstup:

```
4 6
-33  22  47 -42 -18 -35
 13  47   7  10  33  -2
 50 -24 -38  12 -47  -1
  5  27  47  48 -50  39
```

výstup:

```
4
```

Najdlhšia cesta ide postupne cez políčka s výškou 48, 12, 10, 7 a -38.

4. [*20 bodov*] **Knihovník** (programátorská úloha)

Odovzdávanie úlohy: `https://testovac.ksp.sk/tasks/eads2023-du4b/`

Šikovný predavač kníh si všimol, že keď kupujúcemu ponúkne knihu a správne odhadne cenu, tak ju kupujúci kúpi. V stánku na vianočných trhoch má na kope knihy a podľa oblečenia kupujúceho vie odhadnúť, aký je kupujúci prachatý. Ak je kupujúceho prachatosť p a hodnota knihy je h, treba ju kupujúcemu ponúknuť za cenu $p \cdot h$ bubákov a kupujúci ju kúpi.

Najprachatejším kupujúcim chceme tým pádom ponúkať tie najhodnotnejšie knihy. Na trhoch však nie je čas prehrabávať sa všetkými knihami na kope a hľadať tú správnej hodnoty, preto predavač ohodnotí pohľadom kupujúceho a vyberie pre neho **jednu z troch najvrchnejších kníh** na kope.

V stánku máme n kníh o hodnotách $h_1, h_2, ..., h_n$ (hodnoty kníh sú usporiadané od vrchu kopy až po spodok) a v rade stojí n kupujúcich, pričom ich prachatosť je $p_1, p_2, ..., p_n$ (v tomto poradí). Koľko najviac peňazí môže predavač utŕžiť za svoju kopu kníh?

Formát vstupu: V prvom riadku je číslo n. V druhom riadku sú medzerou oddelené hodnoty kníh $h_1, h_2, ..., h_n$. V treťom riadku sú medzerou oddelené prachatosti zákazníkov $p_1, p_2, ..., p_n$. Platí, že $n \leq 500$, $1 \leq h_i \leq 1000$ a $1 \leq p_i \leq 1000$.

Formát výstupu: Na výstupe vypíšte jediné číslo — najväčší možný obnos peňazí, ktorý môžete za knihy utŕžiť.

Príklady:

vstup:

```
4
16 6 2 10
3 8 12 9
```

výstup:

```
336
```

Prvému kupujúcemu (s prachatosťou 3) ponúkneme tretiu knihu z vrchu o hodnote 2, za ktorú zaplatí $3 \cdot 2 = 6$ *bubákov. Zostávajú nám knihy s hodnotami 16 6 10 (v tomto poradí). Druhému kupujúcemu (s prachatosťou 8) ponúkneme druhú knihu z vrchu, za ktorú zaplatí* $8 \cdot 6 = 48$ *bubákov a zostanú nám knihy s hodnotami 16 a 10. Tretí kupujúci zaplatí za vrchnú knihu 192 bubákov a posledný kupujúci zaplatí za vrchnú knihu 90 bubákov.*

Hint 1: Porozmýšľajte nad dynamických programovaním, kde by podproblém reprezentoval stav kopy kníh.
Hint 2: Python je síce fajn jazyk, (a vzorové riešenie v ňom prechádza) ale niekedy je lepšie použiť niečo rýchlejšie. (A Java to nie je)
Hint 3: Váš program na testovači dostane pamäťový limit 256MB, takže ak sa pokúsite naalokovať si väčšie pole/slovník/..., tak vám testovač vráti hlášku `Chyba počas behu programu`. Ak sa tak stane, skúste popremýšľať, že či nemáte naalokovanú časť pamäte, ktorú reálne nepoužívate, resp. vedeli nepoužívať.

Všeobecné pokyny

Písomné úlohy. Písomné úlohy odovzdávajte *do Google Classroom* ako PDF súbory v stanovenom termíne. **Každý príklad odovzdajte v osobitnom PDF súbore.** Na neskoro odovzdané riešenia sa nebude prihliadať. Píšte riešenia takým spôsobom, aby obsahovali všetku potrebnú informáciu na pochopenie vášho riešenia, ale súčasne aby boli stručné a ľahko pochopiteľné. Všetky tvrdenia je potrebné zdôvodniť (a to aj v prípade, že to nie je explicitne napísané v zadaní).

Ak sa v zadaní požaduje vyriešenie algoritmickej úlohy, odovzdajte najlepší algoritmus, aký viete navrhnúť. Základným kritériom na hodnotenie bude *správnosť algoritmu*, druhým kritériom bude jeho *časová, prípadne pamäťová zložitosť*. Správny ale pomalý algoritmus dostane podstatne viac bodov ako algoritmus, ktorý je síce rýchly, ale nedá správnu odpoveď na každý vstup. Neefektívne algoritmy spĺňajúce podmienky zadania dostanú cca 50% bodov. Súčasťou vášho riešenia musia byť nasledujúce časti:

- Najprv popíšte hlavnú myšlienku algoritmu.
- Vyjadrite algoritmus formou pseudokódu.
- Ak to nie je zrejmé na prvý pohľad, ukážte že váš algoritmus je správny.
- Nezabudnite na analýzu zložitosti algoritmu.

Ak nie je povedané inak, logaritmy majú základ 2.

Programátorské úlohy. Pri programátorských úlohách je Vašou úlohou odovzdať len funkčný program, nie je vyžadované písomné riešenie. Riešenie odovzdávate cez webové rozhranie `https://testovac.ksp.sk/tasks/`, kde bude okamžite otestované na niekoľkých vstupoch a dozviete sa, koľko bodov získalo (body získate, keď všetky vstupy z danej sady vyriešite správne v časovom limite). Riešenie môžete odovzdávať aj viackrát, hodnotí sa posledné riešenie odovzdané v stanovenom termíne. Na odovzdávanie riešení (keďže na testovač nefungujú univerzitné prihlasovacie údaje) je nutné sa na stránke zaregistrovať (vľavo na stránke testovača). Pri vytváraní účtu nastavte správne meno a priezvisko, a ako používateľské meno nastavte Váš univerzitný login. Nezabudnite tiež napísať Vaše používateľské meno do PDF súboru k ostatným úlohám, ktoré odovzdávate. Podrobnosti o tom, ako má váš program vyzerať (vrátane povolených programovacích jazykov), nájdete v sekcii "Čo odovzdávať?". Informácie o testovači nájdete v sekcii "Odpovede testovača".

Na zoznámenie sa s rozhraním testovača si môžete vyskúšať naprogramovať niektoré z úloh z časti "Úvod do programovania".