
Záverečná skúška č. 1, 1-AIN-105, Zima 2022/2023

Meno a priezvisko študenta: .....

- Táto písomka má 12 strán. Vyriešte úlohy a napíšte odpovede priamo do zadania. Píšte čitateľne, aby ste zbytočne nestrácali body. Ak vám niekde nestačí miesto, využite prázdne strany.
- Bonusové úlohy budú hodnotené prísnejšie a za nekompletné riešenia nebudú priznávané čiastočné body. Preto ich riešte až na záver, ak vám zostane čas.
- Ak sa vám zadanie zdá nejednoznačné, napíšte predpoklady, ktoré zadanie zjednoznačnia a pokračujte v odpovedi na príslušnú otázku.
- Opisovanie je vážne porušenie akademickej integrity a akýkoľvek prehrešok v tejto oblasti bude postihnutý podľa pravidiel predmetu a postúpený na disciplinárne konanie.

1 (*9 bodov*) **Asymptotická zložitosť pseudokódu**

Ak označíme čas potrebný na vykonanie uvedeného pseudokódu $T(n)$, rozhodnite o pravdivosti nasledujúcich tvrdení. Nie je potrebné žiadne zdôvodnenie.

Upozornenie ohľadom bodovania: Za každú správnu odpoveď dostanete 1 bod, za nesprávnu odpoveď dostanete -1 bod. Nemusíte vyplniť všetky odpovede, nevyplnené odpovede budú za 0 bodov.

a)
```
j:=0; k:=0
for i:=1 to n
  while (j<=i*i)
    print(">")
    j:=j+1
  while (k<=j)
    print("<")
    k:=k+1
```

$T(n) = O(n^3)$ ☐ pravda ☐ nepravda
$T(n) = \Theta(n^3)$ ☐ pravda ☐ nepravda
$T(n) = \Omega(n^3)$ ☐ pravda ☐ nepravda

b)
```
i:=n
while (i>1)
  for j:=1 to i
    print("*")
  i:=2*(i div 3)
```

(n div k je celočíselné delenie)

$T(n) = O(n)$ ☐ pravda ☐ nepravda
$T(n) = \Theta(n)$ ☐ pravda ☐ nepravda
$T(n) = \Omega(n)$ ☐ pravda ☐ nepravda

c)
```
print d(1,n)

function d(i,j):
  if (j-i<1000) return 17
  else
    k:=(j-i+1) div 4;
    a:=d(i,i+2*k)+d(i+1,i+2*k+1)
    b:=d(j-2*k-1,j-1)+d(j-2*k,j)
    for m:=1 to (j-i)*(j-i)
      print("*")
    return a+b;
```

(n div k je celočíselné delenie)

$T(n) = O(n^2)$ ☐ pravda ☐ nepravda
$T(n) = \Theta(n^2)$ ☐ pravda ☐ nepravda
$T(n) = \Omega(n^2)$ ☐ pravda ☐ nepravda

2 (*20 bodov*) **Plátanie hadíc**

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). Cieľom je 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.

a) Úlohu budeme riešiť dynamickým programovaním. Podproblém $H[i]$ bude celková dĺžka záplat, ktoré potrebujeme na zaplátanie dier $d_1, d_2, \ldots, d_i$. Doplňte tabuľku H pre vzorový príklad.

$i = 0$	$i = 1$ $d_1 = 1$	$i = 2$ $d_2 = 3$	$i = 3$ $d_3 = 4$	$i = 4$ $d_4 = 6$	$i = 5$ $d_5 = 10$

b) Profesor Premúdrelý tvrdí, že optimálne riešenie dostaneme aj tak, že v každom kroku položíme ľavý koniec záplaty na najľavejšiu dieru, pričom použijeme 5cm záplatu vždy, ak 5cm záplata priložená na tomto mieste pokryje viac dier ako 3cm záplata. Potom zmažeme zo zoznamu všetky diery pokryté záplatou a opakujeme, až kým neostanú žiadne diery. Ukážte, že profesor Premúdrelý nemá pravdu.

c) Predpokladajme, že v optimálnom riešení bude pokrývať poslednú dieru 5cm záplata tak, že jej pravý okraj presne končí vo vzdialenosti d_k (záplata tak pokrýva dieru na tejto pozícii). Aká je v takom prípade cena optimálneho riešenia? (Môžete použiť hodnoty $H[i]$ pre všetky $i < k$ a hodnoty $d_1, \ldots, d_k$.)

$H[k] =$

d) Napíšte rekurentný vzťah pre výpočet hodnoty $H[i]$ a krátko zdôvodnite jeho správnosť.

$H[i] =$

e) Napíšte pseudokód pre výpočet všetkých hodnôt $H[i]$ pre $i \leq k$ (vrátane výpočtu základných prípadov, ktoré nemožno spočítať vzťahom odvodeným v časti d)). Pseudokód na konci vypíše cenu optimálneho riešenia pôvodnej úlohy pre diery $d_1, \ldots, d_k$.

f) Aká je časová a pamäťová zložitosť vášho riešenia?

3 (*25 bodov*) **Krátke otázky**

Odpovedzte na nasledujúce otázky a krátko (jednou alebo dvoma vetami) zdôvodnite svoju odpoveď.

a) Kruskalov algoritmus v najlepšej implementácii, akú poznáme, spustíme na kružnici s n vrcholmi. Potom jeho časová zložitosť je $\Theta(n^2)$.
Toto tvrdenie je: ☐ pravdivé ☐ nepravdivé
(Označte správnu odpoveď a stručne ju zdôvodnite.)

b) Majme neorientovaný graf, ktorého hrany sú ohodnotené reálnymi číslami v intervale $\langle 0, 1\rangle$. Ohodnotenie hrany reprezentuje pravdepodobnosť jej fungovania. Spoľahlivosť kostry je **súčinom** spoľahlivostí jej jednotlivých hrán. Najspoľahlivejšiu kostru:
☐ možno nájsť v čase $O(mn)$ ☐ je to NP-ťažký problém
(Označte správnu odpoveď a stručne ju zdôvodnite.)

c) Triediaci algoritmus H2020 pracuje tak, že z n prvkovej postupnosti vytvorí 9 postupností dĺžky $n/4$, tieto rekurzívne utriedi a na záver z nich vytvorí výslednú utriedenú postupnosť v čase $O(n)$. Je takýto algoritmus asymptoticky rýchlejší ako Insert Sort, ktorý funguje v čase $\Theta(n^2)$?
☐ áno ☐ nie
(Označte správnu odpoveď a stručne ju zdôvodnite.)

d) Kde sa nachádza v max-halde štvrtý najväčší prvok? (Koreň sa nachádza v hĺbke 0, jeho deti v hĺbke 1, atď.)
☐ vždy v hĺbke 0 ☐ vždy v hĺbke 1 ☐ vždy v hĺbke 2 ☐ vždy v hĺbke 3
☐ ani jedna z predchádzajúcich možností nie je správna
(Označte správnu odpoveď a stručne ju zdôvodnite.)

e) Aká je najhoršia časová zložitosť jedného volania operácie FIND-SET v dátovej štruktúre UNION/FIND-SET, pokiaľ ju implementujeme s použitím rankov, ale bez kompresie cesty?
☐ $\Theta(n \log n)$ ☐ $\Theta(n)$ ☐ $\Theta(\log n)$ ☐ $\Theta(1)$
☐ ani jedna z predchádzajúcich možností nie je správna
(Označte správnu odpoveď a stručne ju zdôvodnite.)

4 (*26 bodov*) **Kde je chyba?**

Každý z nasledujúcich dôkazov obsahuje fatálnu logickú chybu. Vysvetlite, kde je chyba.

a) Zostavíme polynomiálny algoritmus pre hľadanie dĺžky najkratšej jednoduchej cesty v grafe so zápornými hranami. (Jednoduchá cesta je taká, v ktorej sa žiadny vrchol nevyskytuje viackrát.)

Nech najmenšia dĺžka hrany v našom grafe je $W < 0$. Zmodifikujeme graf tak, že od ohodnotenia každej hrany odčítame W a na takto zmodifikovaný graf, ktorý nemá žiadne záporné hrany, spustíme Dijkstrov algoritmus. Ku výslednej najkratšej ceste pripočítame kW, kde k je počet jej hrán, a dostaneme dĺžku najkratšej cesty v pôvodnom grafe. Keďže najmenšiu dĺžku hrany vieme nájsť v čase $O(m)$ a Dijkstrov algoritmus beží v čase $O(n^2)$, tak aj náš výsledný algoritmus bude bežať v čase $O(n^2)$.

b) Problém 2-SAT je problém rozhodnuteľnosti logických formúl v tvare $c_1 \wedge c_2 \wedge \ldots \wedge c_m$, kde každá klauzula c_i je disjunkciou dvoch literálov (napr. $x_1 \vee \neg x_6$).

Budeme dokazovať, že problém 3-SAT je NP-ťažký. Nech $f = c_1 \wedge c_2 \wedge \ldots \wedge c_m$ je 2-SAT formula s m klauzulami nad premennými $x_1, \ldots, x_n$. Pridáme novú premennú x_{n+1}. Potom formula $f' = (c_1 \vee x_{n+1}) \wedge \ldots \wedge (c_m \vee x_{n+1}) \wedge (c_1 \vee \neg x_{n+1}) \wedge \ldots \wedge (c_m \vee \neg x_{n+1})$ má $2m$ klauzúl, pričom každá klauzula má tri literály, čiže je to inštancia problému 3-SAT. Súčasne platí, že formula f je splniteľná práve vtedy, keď je splniteľná f'.

Teda sme ukázali polynomiálnu redukciu 2-SAT$\leq_P$3-SAT, a teda problém 3-SAT je NP-ťažký problém.

5 (*20 bodov*) **Sufixové stromy**

a) Nakreslite sufixový strom slova `HATLAPATLA`. Pripomeňme si, že sufixový strom je **komprimovaný** trie, v ktorom sú uložené všetky sufixy príslušného reťazca.
(Nezabudnite, že pri budovaní sufixového stromu vždy pripojíme na koniec špeciálny znak `$`, ktorý sa nevyskytuje nikde inde.)

b) Predpokladajme, že máte **nekomprimovaný** trie, v ktorom sú uložené všetky sufixy nejakého reťazca T (t.j. je to v zásade sufixový strom ale bez kompresie hrán). **Napíšte a okomentujte pseudokód** pre funkciu, ktorá pomocou tohto trie vypočíta počet výskytov reťazca P v reťazci T.

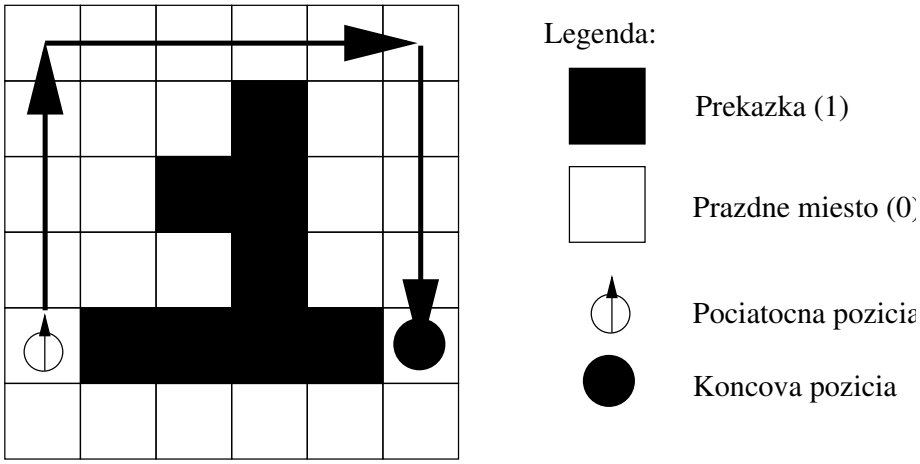
Príslušná funkcia `pocetVyskytov(koren,P)` dostane na vstupe reťazec P a koreň nekomprimovaného trie `koren` reprezentujúceho reťazec T. V každom vrchole trie x (vrátane koreňa) je uložené pole tr, pričom $x.\mathrm{tr}[c]$ ukazuje na dieťa vrcholu x, do ktorého sa dostaneme keď použijeme znak c. Ak také dieťa neexistuje, hodnota je `None`.

6 (*20 bodov*) **Bonusová úloha: Plánovanie pohybu robota**

Uvažujme štvorcovú miestnosť reprezentovanú maticou $A[1..n, 1..n]$, v ktorej nuly znamenajú prázdne miesto a jednotky znamenajú prekážky. Robot sa nachádza na počiatočnej pozícii (i, j) a je otočený na sever a chceme, aby sa dostal na cieľovú pozíciu (k, l) (nezáleží na tom, ako bude robot na konci otočený).

Robot sa dokáže pohybovať veľmi rýchlo po priamke: za jednu sekundu sa dokáže premiestniť na ľubovoľnú pozíciu v miestnosti v smere, ktorým je otočený, a to bez ohľadu na vzdialenosť, pokiaľ pri tom nemusí preraziť prekážku (cez prekážky sa prejsť nedá, lebo sú tvrdšie ako robot). Robot sa však otáča veľmi pomaly: na otočenie o 90 stupňov doľava alebo doprava potrebuje tiež jednu sekundu. Napíšte čo najefektívnejší algoritmus, ktorý naplánuje pre robota najrýchlejšiu cestu (najrýchlejšia cesta v tomto prípade nemusí byť to isté ako najkratšia cesta). Zdôvodnite správnosť algoritmu a odhadnite jeho časovú a pamäťovú zložitosť.

Príklad:



Robot sa za prvú sekundu presunie zo svojej štartovacej pozície na pozíciu $(1, 1)$. Za druhú sekundu sa otočí doprava, tretiu sekundu strávi presunom na pozíciu $(1, 6)$, za štvrtú sekundu sa opäť otočí doprava a za piatu sekundu dôjde do cieľa, pričom zostane otočený na juh. Toto je najrýchlejšia cesta pre tento vstup.

(Táto strana je naschvál prázdna, môžete sem písať riešenia, ale **zaznačte to pri zadaní úlohy**)

(Táto strana je naschvál prázdna, môžete sem písať riešenia, ale **zaznačte to pri zadaní úlohy**)