Problémy

Navrhněte algoritmus, který v zadané posloupnosti čísel $a_1, a_2, \ldots, a_n$ najde

① maximum i minimum za použití nejvýše $3\lfloor n/2 \rfloor$ porovnání

② druhé největší číslo za použití nejvýše $n\text{-}2+\lceil \log_2 n \rceil$ porovnání

Problémy ✎

Dokažte nebo vyvraťte:

Pro každou dvojici funkcí f,g: $\mathbf{N} \rightarrow \mathbf{R}^+$ platí

① pokud $f(n)=O(g(n))$, pak $g(n)=O(f(n))$

② pokud $f(n)=O(g(n))$, pak $2^{f(n)}=O(2^{g(n)})$

③ pokud $f(n)=O(g(n))$, pak $g(n)=\Omega(f(n))$

④ $f(n)=O(f(n)^2)$

Problémy

① Vylepšete pythonovskou funkci sito(n) pro generování prvočísel metodou Erastothenova síta tak, aby v seznamu `je_prv[]` nebyla evidována sudá čísla > 2.

② Srovnete složitost Euklidova algoritmu se složitostí algoritmu výpočtu NSD pomocí rozkladu na prvočinitele.

③ Navrhněte efektivní algoritmus výpočtu nejmenšího společného násobku dvou zadaných přirozených čísel.

Problémy ✎

① Zobecněte funkce `bin2dec` a `dec2bin` tak, aby prováděly konverzi z / do libovolné číselné soustavy o základu b, $2 \leq b \leq 16$. Je-li $b > 10$, chybějící cifry reprezentujte velkými písmeny ze začátku abecedy, tj.

`A, B, C, D, E, F.`

② V jazyce Python navrhnete funkci

soucet(a,b) ,

která vrátí součet dvou čísel, zadaných seznamem svých cifer. Zvažte obě varianty pořadí (od nejvýznamějšího / od nejméně významného řádu).

Problémy 

① V jazyce Python sestavte funkci

`heapSort(a)`

která setřídí prvky zadaného pole `a` vzestupně haldovým tříděním. Váš algoritmus byl měl třídit na místě, tj. může využívat jen konstantní pracovní paměť.

Problémy ✎

② Uvažte třídící algoritmus, který pro libovolné dva prvky a_i, a_j na vstupu může na dotaz a_i ? a_j obdržet jednu ze tří možných odpovědí:

- $a_i < a_j$
- $a_i > a_j$
- $a_i = a_j$

Bude náš dolní odhad platit i tomto případě?

Problémy ✎

① Zobecněte funkci odmocnina(n) tak, aby fungovala pro libovolné kladné číslo n.

② Na přednášce jsme zjistili, že k uhodnutí myšleného přirozeného čísla x, $1 \leq x \leq n$, stačí $\lceil \log_2 n \rceil + 1$ pokusů (odpovědi: uhodnuto/větší/menší, Hi-Lo Game). Zobecněte řešení na případ, kdy x může být libovolné přirozené číslo. Lze je uhodnout položením $O(\log x)$ dotazů?

Problémy: spojové seznamy

① Do implementace zásobníku a fronty pomocí spojových seznamů doplňte ošetření chybových stavů.

② Rozmyslete si realizaci dalších operací nad spojovými seznamy:

- ulož prvek na určené místo
- odeber prvek z určeného místa
- najdi prvek s max / min klíčem
- obrácení seznamu
- zřetězení dvou seznamů

✎ Problém: prioritní fronta

② Pro ADT prioritní fronta navrhněte operace :

- ZvýšeníPriority zadaného prvku
- Odstraň zadaný prvek

Zvládneme realizovat obě operace v čase O(log n) ?

Můžete předpokládat, že odkaz (index do pole při implementaci binární haldou) na příslušný prvek je zadán na vstupu, do časové složitosti obou operací tedy nemusíte zahrnovat čas potřebný k jeho vyhledání.

Problém: rekurze vs. iterace

1. V jazyce Python sestavte funkce
   - pro výpočet funkce $n!$
   - pro výpočet n-tého prvku Fibonacciho posloupnosti

Pro každý problém se pokuste navrhnout dvě řešení

- rekurzivní
- iterativní (bez rekurze)

a porovnat jejich efektivitu.

✎ Problémy: Hanojská věž

② Kolik tahů potřebuje náš (rekurzivní) algoritmus na přemístění n kotoučů?

③ Dle legendy existuje v Asii chrám, v němž každý den v poledne mniši slavnostně přemístí jeden z 64 zlatých kotoučů. Jakmile bude přemístěna celá “věž”, nastane konec světa. Spočítejte, za jak dlouho k tomu může dojít.

✎ Problémy: Binární stromy

① V jazyce Python navrhněte funkci, která obdrží binární strom a spočítá jeho

- výšku
- průměrnou výšku

Přitom výška (průměrná výška) stromu je definována jeko maximání (průměrná) délka cesty z kořene do listu.

② Navrhněte efektivní algoritmus, který zjistí, zdali je zadaný binární strom symetrický dle svislé osy, procházející jeho kořenem.