Příklad 1: Mějme čtvercovou mřížku (tabulku čokolády) 4×8. Jeden lom pokryje libovolně dlouhý rovný úsek hranice mezi čtverci, nepřerušený jiným lomem. Kolik lomů potřebujeme pro pokrytí všech hranic?

Pojmy: invariant, potenciál

RAM

Příklad 2: Napiště Euklidův algoritmus na výpočet GCD jako RAM kód.

Příklad 3: Napiště pseudokód pro bubble-sort a převeďte ho na RAM kód.

Binární vyhledávání

Příklad 4: Základ Jak vypadá obecné binární vyhledávání? Co je třeba zajistit, aby bylo prokazatelně konečné a korektní?

Příklad 5: Násobnost V poli máme n vzestupně seřazených čísel $x_1, \ldots, x_n$. Jak upravit binární vyhledávání tak, aby vrátilo indexy prvního a posledního výskytu pro dané číslo k?

Příklad 6: Funkce Pro (nezáporné) číslo n na vstupu najděte hodnotu $\lfloor \sqrt{n} \rfloor$ pomocí základních aritmetických operací.

Příklad 7: Neomezené vyhledávání Mějme neklesající funkci f. Jak efektivně najdeme nejmenší přirozené číslo m t.ž. $f(m) = c$? Nemáme žádný horní odhad na m nebo růst f. Chceme časovou složitost omezenou funkcí v m.

Příklad 8: Rychlý odhad Uvažujme stejnou situaci jako v předchozím příkladě. Definujme multiplikativní odhad k jako $2^{k-1} \leq m < 2^k$. Jak můžeme najít k v čase $\mathcal{O}(loglog(m))$? Můžeme s pomocí tohoto postupu asymptoticky zrychlit náš algoritmus pro nalezení m?

Datové struktury

- Jaké známe základní datové struktury?
- Jak měříme složitost struktur?
- Jak jsou definované typy datových struktur? (např. co je obecně halda?)
- Je vyhledávací strom halda?

Příklad 9: Máme matici velikosti $n \times n$. Chceme najít její největší nulovou podmatici. Lze to provést v lineárním čase?

Příklad 1: Mějme čtvercovou mřížku (tabulku čokolády) 4×8. Jeden lom pokryje libovolně dlouhý rovný úsek hranice mezi čtverci, nepřerušený jiným lomem. Kolik lomů potřebujeme pro pokrytí všech hranic?

Pojmy: invariant, potenciál

RAM

Příklad 2: Napiště Euklidův algoritmus na výpočet GCD jako RAM kód.

Příklad 3: Napiště pseudokód pro bubble-sort a převeďte ho na RAM kód.

Binární vyhledávání

Příklad 4: Základ Jak vypadá obecné binární vyhledávání? Co je třeba zajistit, aby bylo prokazatelně konečné a korektní?

Příklad 5: Násobnost V poli máme n vzestupně seřazených čísel $x_1, \ldots, x_n$. Jak upravit binární vyhledávání tak, aby vrátilo indexy prvního a posledního výskytu pro dané číslo k?

Příklad 6: Funkce Pro (nezáporné) číslo n na vstupu najděte hodnotu $\lfloor \sqrt{n} \rfloor$ pomocí základních aritmetických operací.

Příklad 7: Neomezené vyhledávání Mějme neklesající funkci f. Jak efektivně najdeme nejmenší přirozené číslo m t.ž. $f(m) = c$? Nemáme žádný horní odhad na m nebo růst f. Chceme časovou složitost omezenou funkcí v m.

Příklad 8: Rychlý odhad Uvažujme stejnou situaci jako v předchozím příkladě. Definujme multiplikativní odhad k jako $2^{k-1} \leq m < 2^k$. Jak můžeme najít k v čase $\mathcal{O}(loglog(m))$? Můžeme s pomocí tohoto postupu asymptoticky zrychlit náš algoritmus pro nalezení m?

Datové struktury

- Jaké známe základní datové struktury?
- Jak měříme složitost struktur?
- Jak jsou definované typy datových struktur? (např. co je obecně halda?)
- Je vyhledávací strom halda?

Příklad 9: Máme matici velikosti $n \times n$. Chceme najít její největší nulovou podmatici. Lze to provést v lineárním čase?

Domácí úkol 1: Mezery

Mějme na vstupu posloupnost kladných, navzájem různých celých čísel, seřazených vzestupně. Pro konstantu k na vstupu navrhněte algoritmus, který nalezne k-té chybějící číslo. Chceme optimálně rychlý algoritmus.

Protože uvažujeme všechna kladná celá čísla, pokud posloupnost nezačíná číslem 1, je 1 první chybějící číslo.

Předpoklady: Hodnota k je kladné celé číslo. Vstup je reprezentován jako **pole n hodnot indexovaných** $1, \ldots, n$ a dvě proměnné k a n.

Příklady vstupů:

- $k = 2$,$n = 5$, data: $1, 3, 4, 5, 7$, odpověď: 6
- $k = 2$,$n = 3$, data: $4, 5, 6$, odpověď: 2
- $k = 2$,$n = 5$, data: $1, 2, 3, 4, 5$, odpověď: 7

Plnohodnotné řešení by mělo obsahovat

- Stručný popis metody řešení (není slovní pseudokód)
- Pseudokód (nízkoúrovňový, stylu "hezkého" RAM)
- Ošetření speciálních případů
- Důkaz korektnosti
- Analýza časové složitosti (se zdůvodněním)

Pro zápis pseudokódu můžete používat veškeré přirozené konstrukce, které lze na RAM "simulovat" s konstantním zpomalením. Tj. smíte používat proměnné, přiřazení s komplexními výpočty, cykly, if-else podmínky s logickými výrazy, vrátit výstup pomocí return, ...

Přezdívka:

Výsledky budou zveřejňovány na webu cvičení pod přezdívkami. Zvolte si přezdívku nebo udejte explicitní souhlas s užitím jména na místo přezdívky. Chybějící přezdívky budou vygenerovány.

Domácí úkol 1: Mezery

Mějme na vstupu posloupnost kladných, navzájem různých celých čísel, seřazených vzestupně. Pro konstantu k na vstupu navrhněte algoritmus, který nalezne k-té chybějící číslo. Chceme optimálně rychlý algoritmus.

Protože uvažujeme všechna kladná celá čísla, pokud posloupnost nezačíná číslem 1, je 1 první chybějící číslo.

Předpoklady: Hodnota k je kladné celé číslo. Vstup je reprezentován jako **pole n hodnot indexovaných** $1, \ldots, n$ a dvě proměnné k a n.

Příklady vstupů:

- $k = 2$,$n = 5$, data: $1, 3, 4, 5, 7$, odpověď: 6
- $k = 2$,$n = 3$, data: $4, 5, 6$, odpověď: 2
- $k = 2$,$n = 5$, data: $1, 2, 3, 4, 5$, odpověď: 7

Plnohodnotné řešení by mělo obsahovat

- Stručný popis metody řešení (není slovní pseudokód)
- Pseudokód (nízkoúrovňový, stylu "hezkého" RAM)
- Ošetření speciálních případů
- Důkaz korektnosti
- Analýza časové složitosti (se zdůvodněním)

Pro zápis pseudokódu můžete používat veškeré přirozené konstrukce, které lze na RAM "simulovat" s konstantním zpomalením. Tj. smíte používat proměnné, přiřazení s komplexními výpočty, cykly, if-else podmínky s logickými výrazy, vrátit výstup pomocí return, ...

Přezdívka:

Výsledky budou zveřejňovány na webu cvičení pod přezdívkami. Zvolte si přezdívku nebo udejte explicitní souhlas s užitím jména na místo přezdívky. Chybějící přezdívky budou vygenerovány.