**1.** Kolika různými způsoby lze uzávorkovat součin matic? (Různé způsoby závorkování odpovídají různému průběhu výpočtu a obecně různým mezivýsledkům. Závorkování (X) a ((X)) pokládáme za totožné.)

a) A × B × C × D

b) A × B × C × D × E

**2.** Předpokládejme, že na vypočtení součinu matic A × B je zapotřebí *r*∙*s*∙*t* operací, kde A ∈ **R***r*×*s* a B ∈ **R***s*×*t.*

Určete, kolik operací je třeba na výpočet součinu (A × B) × C a kolik na výpočet součinu A × (B × C), pokud

a) A ∈ **R**2×3 , B ∈ **R**3×5 , C ∈ **R**5×4

b) A ∈ **R**3×4 , B ∈ **R**4×5 , C ∈ **R**5×2

c) A ∈ **R***n*×4 , B ∈ **R**4×2*n* , C ∈ **R**2*n*×3

**3.** Určete, pro které hodnoty *n* je výhodnější vypočítat součin (A × B) × C než vypočítat součin A × (B × C).

a) A ∈ **R***n*×2 , B ∈ **R**2×3 , C ∈ **R**3×4

b) A ∈ **R**5×*n* , B ∈ **R***n*×4 , C ∈ **R**4×*n*

c) A ∈ **R***n*×*n* , B ∈ **R***n*×100 , C ∈ **R**100×*n*

**4.** Rozměry matic A, B, C, D, E, jsou po řadě 2 × 5, 5 × 3, 3 × 6, 6 × 2, 2 × 4. Určete metodou dynamického programování, jak uzávorkovat součin A × B × C × D × E, aby počet operací násobení dvou čísel během výpočtu celého součinu byl co nejmenší. Kolik to bude operací?

**5.** Rozměry matic A, B, C, D, E, jsou stejné jako v předchozí úloze. Z důvodů např. testování HW/SW si přejeme součin uzávorkovat tak, aby počet operací násobení dvou čísel byl co největší. Rozhodněte, zda lze metodu hledání co nejvýhodnějšího uzávorkování přizpůsobit pro řešení této modifikované úlohy. Jaký bude algoritmus řešení?

**6.**  Použijte myšlenku výpočtu optimálního uzávorkování v součinu matic pro řešení jednodušší úlohy:

Kolika různými způsoby lze uzávorkovat součin *n* matic? Budete potřebovat 2D nebo 1D tabulku? Ověřte výsledek pro několik malých hodnot *n*, měl by vyjít .

**7.** Najděte nejdelší rostoucí podposloupnost dané posloupnosti. Použijte metodu dynamického programování, napište tabulku průběžných délek částečných výsledků a tabulku

předchůdců.

a) 5 8 11 13 9 4 1 2 0 3 7 10 12 6

b) 6 7 5 15 10 9 11 18 19 8 12 1 3 4 13 14 0 17 2 16

**8**. Určete, jak je možno využít nebo přizpůsobit metodu hledání nejdelší rostoucí podposloupnosti pro hledání

a) nejdelší klesající podposloupnosti

b) nejdelší neklesající podposloupnosti

c) nejdelší konstantní podposloupnosti

d) nejdelší alternující posloupnosti.

V případě c) uvažte, zda existuje asymptoticky rychlejší metoda než dynamické programování.

V případě d) v alternující posloupnosti a[1], a[2], ... a[n] platí

a[k] − a[k−1])\* (a[k+1] − a[k]) < 0, pro k = 2, 3, ..., *n*−1.

**9**. Řetězec x měníme na řetězec y tak, že postupně äplikujeme operace Insert, Delete, Rewrite v libovolném pořadí na jeden z řetězců. Rewrite = přepiš libovolný znak na jiný, Insert = vlož libovolný znak na libovolné místo, Delete = smaž libovolný znak. Editační vzdálenost dvou řetězců je rovna minimálnímu počtu operací, které jeden řetězec promění v druhý. Určete počet všech řetězců, které mají

 a) od řetězce aba vzdálenost 1,

 b) od řetězce aabbaa vzdálenost 2.

Používat lze jen znaky 'a' a 'b'.

**10.**  Minimální počet editačních operací(Insert, Delete, Rewrite) potřebných k přepsání řetězce X na řetězec Y dán prvkem T[|X|] [|Y|] tabulky T, pro jejíž obsah platí:

T[0][k] = k, T[i][0] = i;

T[i][k] = min{T[i−1][k], T[i][k−1], T[i−1][k−1] + (X[i]=Y[k])? 0:1}

Jaká je editační vzdálenost slov

00110011, 012012012, abcdabcdaadda, dacbdbbc?

**11.**  Navrhněte dodatečné značky, které budte umisťovat do tabulky T registrující editační vzdálenost dvou řetězců. S použitím těchto značek rekonstruujte v čase O(max(|*x*|,|*y*|)) editační operacepotřebné k přepsání řetězce *x* na řetězec *y*. Inspirujte se tabulkou pro určování nejdelší společné podposloupnosti.