

ALGORYTMY I STRUKTURY DANYCH

IIUWr. II rok informatyki.

1. (0pkt) Rozwiąż wszystkie zadania dodatkowe.
2. (1pkt) Zmień podany na wykładzie algorytm znajdujący najtańszą drogę przejścia przez tablicę tak, by znajdował drogę o drugim co do wielkości koszcie.
3. (2pkt) Rozważmy następujące operacje na ciągach:
 - $insert(x, i, a)$ - wstawienie a pomiędzy i -tym i $(i + 1)$ -szym elementem x -a;
 - $delete(x, i)$ - usunięcie i -tego elementu x -a;
 - $replace(x, i, a)$ - zastąpienie i -tego elementu x -a przez a .

Jak łatwo zauważyć, dla każdych dwóch ciągów x i y istnieją sekwencje powyższych operacji przekształcające x w y . Jeśli każdej operacji przypiszemy koszt (nieujemną liczbę rzeczywistą) możemy mówić o minimalnym koszcie przekształcenia x w y (koszt ten nazywa się *odległością edycyjną* ciągów x i y).

Ułóż algorytm, który dla danych dwóch ciągów znajdzie ich odległość edycyjną.

4. (2pkt) Na trzelementowym zbiorze $A = \{a, b, c\}$ określono operację \odot . Nie jest ona ani przemienna ani łączna. Ułóż algorytm, który dla danego ciągu $x_1 \odot x_2 \odot \dots \odot x_n$, gdzie x_i są symbolami ze zbioru A , rostrzyga, czy można w nim tak rozstawić nawiasy, by wartość otrzymanego wyrażenia wynosiła a .
5. (2pkt) Dany jest graf pełny $G = (V, E)$ z nieujemnymi wagami na krawędziach oraz ciąg wszystkich jego wierzchołków $C = \langle v_1, \dots, v_n \rangle$. Początkowo w wierzchołku v_1 znajdują się dwa pionki. W kolejnych ruchach masz przesunąć pionki według następujących zasad:
 - w każdym ruchu przesuwasz jeden pionek,
 - pionek stojący w wierzchołku v_i możesz być przesunąć do wierzchołka v_j jedynie wtedy, gdy $j > i$ (czyli do wierzchołka znajdującego się dalej w ciągu C),
 - wszystkie wierzchołki grafu muszą być odwiedzone przez co najmniej jeden pionek,
 - po ostatnim ruchu obydwaj pionki znajdują się w wierzchołku v_n .

Ułóż algorytm obliczający ciąg ruchów pionków minimalizujący sumę długości dróg przebytych przez pionki (przez długość drogi rozumiemy sumę wag jej krawędzi).

6. (1 pkt) Rozważmy następujący problem *3-podziału*. Dla danych liczb całkowitych $a_1, \dots, a_n \in \langle -C..C \rangle$ chcemy stwierdzić, czy można podzielić zbiór $\{1, 2, \dots, n\}$ na trzy rozłączne podzbiory I, J, K , takie, że

$$\sum_{i \in I} a_i = \sum_{j \in J} a_j = \sum_{k \in K} a_k.$$

7. (2pkt) Ułóż algorytm rozwiązujący poniższy problem triangulacji wielokąta wypukłego:

PROBLEM:

Dane: ciąg par liczb rzeczywistych $(x_1, y_1), \dots, (x_n, y_n)$, określających kolejne wierzchołki n -kąta wypukłego P

ZAŁOŻENIE: dane są określone poprawnie.

Zadanie: Znaleźć zbiór S nieprzecinających się przekątnych, które dzielą P na trójkąty, taki, że długość najdłuższej przekątnej w S jest możliwie najmniejsza.

ZADANIA DODATKOWE - NIE BĘDĄ ROZWIĄZYWANE W CZASIE ĆWICZEŃ

1. (1pkt) Uzupełnij podany na wykładzie algorytm sprawdzający przynależność słowa do języka generowanego przez bezkontekstową gramatykę w normalnej postaci Chomsky'ego tak, by w przypadku pozytywnej odpowiedzi wypisywał jego wyprowadzenie.
2. (1pkt) Jak zmieni się złożoność problemu przynależność słowa do języka generowanego przez bezkontekstową gramatykę w normalnej postaci Chomsky'ego, jeśli gramatyka także będzie daną wejściową?
3. (2pkt) *Gramatyką liniową* nazywamy gramatykę bezkontekstową, w której prawe strony produkcji zawierają co najwyżej jeden symbol nieterminalny. Ułóż algorytm sprawdzający przynależność słowa do języka liniowego, który wykorzystuje pamięć rozmiaru $O(n)$.
4. (2pkt) Dana jest szachownica $n \times n$ i pozycje pionów na niej (może być ich nawet $O(n^2)$). Ułóż algorytm znajdujący prostokątny fragment szachownicy o największym polu, na którym nie znajduje się ani jeden pion. Twój algorytm powinien działać w czasie $O(n^2)$.
5. (2pkt) Napisz w pseudopascalu lub pseudoC++ dwie procedury:
 - (a) drukującą ciąg nazw macierzy wraz z poprawnie rozstawionymi nawiasami wyznaczającymi optymalną kolejność mnożenia macierzy,
 - (b) drukującą ciąg instrukcji postaci $A \leftarrow B \times C$, prowadzących do obliczenia w optymalny sposób iloczynu macierzy (A jest nazwą macierzy roboczej, a B i C - są nazwami macierzy wejściowych lub wcześniej obliczonych macierzy roboczych).
6. (1pkt) Udowodnij, że liczba wywołań rekurencyjnych w poniższej procedurze obliczającej minimalny koszt pomnożenia macierzy jest $\Theta(3^n)$.


```

function minmat( $i, j$ )
  if  $i = j$  then return 0
   $ans \leftarrow \infty$ 
  for  $k \leftarrow i$  to  $j - 1$  do
     $ans \leftarrow \min(ans, d_{i-1}d_kd_j + \text{minmat}(i, k) + \text{minmat}(k + 1, j))$ 
  return  $ans$ 
      
```
7. (2pkt) Jak wiesz, liczba wszystkich poprawnych rozstawień n par nawiasów (a więc i sposobów pomnożenia n macierzy) jest równa n -tej liczbie Catalana.
 - Wykaż, że liczba ta rośnie szybciej niż 3^n .
 - Czy potrafisz wskazać poprawne rozstawienie nawiasów, które nie jest rozważane przez procedurę *minmat*?
8. (2pkt) Dany jest zbiór n przedmiotów $A = \{a_1, \dots, a_n\}$. Dla każdego przedmiotu znamy jego wagę $w(a_i)$ oraz jego cenę $c(a_i)$; obie te liczby są naturalne a $c(a_i)$ jest nie większe od n^2 . Dana jest ponadto liczba naturalna P . Ułóż algorytm znajdujący podzbiór S zbioru przedmiotów, taki, że suma wag przedmiotów z S nie przekracza P , a suma ich cen jest możliwie największa. W jakim czasie działa Twój algorytm?
9. (1pkt) Pokaż jak obliczyć długość elementów *LCS* używając jedynie $2 \min(m, n)$ -elementowej tablicy c plus $O(1)$ dodatkowej pamięci. Następnie pokaż, jak to zrobić używając $\min(m, n)$ -elementowej tablicy c plus $O(1)$ dodatkowej pamięci
10. (2pkt) Zmodyfikuj algorytm znajdujący najdłuższy wspólny podciąg dwóch ciągów n elementów, tak by działał w czasie $O(n^2)$ i używał $O(n)$ pamięci.

Rysunek 1: *Podwójna drabina n -elementowa*

11. (2pkt) Podwójną drabiną rozmiaru n nazywamy graf przedstawiony na poniższym rysunku.

Uogólnij na podwójne drabiny podany na wykładzie algorytm znajdujący liczbę drzew rozpinających o k krawędziach wyróżnionych.

Krzysztof Loryś