Egzamin z ASD – II termin
21.II.2022

Zadanie 1 [7 punktów]

W tablicy $P[1..n]$ opisano n-wierzchołkowe drzewo z korzeniem. Wierzchołki drzewa ponumerowano liczbami 1, 2, …, n i każdy wierzchołek jest identyfikowany z jego numerem. Jeżeli wierzchołek v nie jest korzeniem , to $P[v]$ jest rodzicem wierzchołka v w drzewie. Wierzchołek v jest korzeniem wtedy i tylko wtedy, gdy $P[v] = v$. Na początku wszystkie wierzchołki są czarne. Na drzewie wykonujemy następujące operacje:

Przekoloruj(v,k):: każdy z czarnych wierzchołków spośród v, $P[v]$, $P[P[v]]$, …, $P^k[v]$ zamień na biały ($k \geq 0$, $P^0[v] = v$),
DłBiałej(v):: podaj długość najdłuższej białej ścieżki o początku w wierzchołku v w kierunku do korzenia (wierzchołki na ścieżce nie mogą się powtarzać).

Zaproponuj strukturę danych, która umożliwi wydajne wykonanie ciągu m > n operacji *Przekoloruj* i *DłBiałej* przy założeniu, że kolejność wykonywania tych operacji może być dowolna.

Zadanie 2 [10 punktów]

Powiemy, że graf nieskierowany G jest *prawie drzewem* wtedy i tylko wtedy, gdy jest spójny i każda jego krawędź należy do co najwyżej jednego cyklu elementarnego.

a) **[6 punktów]** Załóżmy, że G jest grafem ważonym o dodatnich wagach na krawędziach. Zaproponuj wydajny algorytm, który dla danego źródła s policzy wagi najlżejszych ścieżek prowadzących z s do wszystkich pozostałych wierzchołków.
b) **[4 punkty]** Powiemy, że prawie drzewo jest *maksymalne* gdy, każda krawędź należy do jakiegoś cyklu elementarnego.
    1. Ile wynosi minimalna a ile maksymalna wysokość BFS drzewa w maksymalnym n-wierzchołkowym prawie drzewie?
    2. Ile wynosi minimalna a ile maksymalna wysokość DFS drzewa w maksymalnym n-wierzchołkowym prawie drzewie?

Uwaga: cyklem elementarnym nazywamy cykl o długości co najmniej 3, którego wierzchołki są parami różne. Ponadto możesz przyjąć, że $n \geq 3$.

Zadanie 3 [9 punktów]

Dokonaj analizy następującego, rekurencyjnego algorytmu sortowania tablicy $a[1..n]$, dla $n > 0$.

```
Sort(i,j){
// 1 ≤ i ≤ j ≤ n
  if (i + 1) = j  then if a[i] > a[j] then  a[i] :=: a[j]; // zamiana wartości zmiennych
  if (i+1) < j then{
    s := ⌈2(j − i + 1)/3⌉;
    Sort(i,i+s-1); Sort(j-s+1,j); Sort(i,i+s-1)
  }
}
```

a) **[2 punkty]** Udowodnij, że wywołanie *Sort*(1,n) posortuje tablicę a niemalejąco.

b) **[3 punkty]** Ile razy zamiana a[i] :=: a[j] zostanie dokonana dla n = 4k i a[1..n] = [n, 1, n-2, 3, …, 2, n-1]?
c) **[4 punkty]** Dokonaj analizy pesymistycznej złożoności *Sort*(1,*n*) ze względu na liczbę porównań pomiędzy elementami tablicy *a*.

Zadanie 4 [14 punktów]

W tym zadaniu rozważamy niepuste słowa nad alfabetem {a,b} o długości $n > 2$, w których pierwszym symbolem jest zawsze 'a'.

a) **[2 punkty]** Dla ilu słów o początku 'ab' każda z wartości w tablicy prefiksów-sufiksów jest nie większa od 1?
b) **[4 punkty]** Dla ilu słów o początku 'aa' każda z wartości w tablicy prefiksów-sufiksów jest nie większa od 1?
c) **[8 punktów]** Zaproponuj wydajny algorytm, który dla danego słowa *x* o długości *n* obliczy tablicę $W[1..n]$ taką, że $W[i]$ jest liczbą wszystkich różnych podsłów w sufiksie $x[i..n]$.

Uzasadnij poprawność swoich rozwiązań i dokonaj analizy złożoności obliczeniowej zaproponowanych algorytmów.