[English version is on the second page.]

Udowodnij, że mając dany graf skierowany oraz pewien porządek na jego wierzchołkach, można sprawdzić w deterministycznej pamięci logarytmicznej czy ten porządek to porządek DFS.

Dokładniej, rozważamy następujący algorytm DFS:

```
function DFS(v)
    odwiedzony[v] ← True
    ostNr ← ostNr + 1
    nr[v] ← ostNr
    for all w ∈ Następniki(v) do        ▷ następniki v są rozważane w nieustalonej kolejności
        if not odwiedzony[w] then
            DFS(w)
        end if
    end for
end function
ostNr ← 0
for all v ∈ Wierzchołki do
    odwiedzony[v] ← False
end for
for all v ∈ Wierzchołki do        ▷ wierzchołki grafu są rozważane w nieustalonej kolejności
    if not odwiedzony[v] then
        DFS(v)
    end if
end for
```

Niech *DFS_Order* będzie językiem słów postaci $n\$macierz\$porządek$ takich, że dla pewnego grafu skierowanego G i dla pewnego biegu powyższego algorytmu (czyli dla pewnej kolejności rozważania wierzchołków w pętlach „for”),

- n jest liczbą wierzchołków G, zapisaną binarnie,
- *macierz* jest napisem składającym się z n^2 bitów, gdzie 1 na pozycji $n \cdot (i-1) + j$, dla $1 \leq i, j \leq n$, oznacza, że istnieje krawędź z wierzchołka i do wierzchołka j, natomiast 0 oznacza, że nie ma takiej krawędzi (nie może być krawędzi z i do i),
- *porządek* to ciąg n liczb zapisanych binarnie, pooddzielanych znakiem \$, gdzie i-ta liczba jest wartością przypisaną przez algorytm do $nr[i]$.

Celem jest udowodnienie, że *DFS_Order* jest w **L**.

Prove that, given a directed graph and an ordering of its nodes, it is possible to check in deterministic logarithmic space whether this ordering is an DFS ordering.

More precisely, we consider the following DFS algorithm:

```
function DFS(v)
    visited[v] ← True
    lastNr ← lastNr + 1
    nr[v] ← lastNr
    for all w ∈ Successors(v) do          ▷ successors of v are taken in random order
        if not visited[w] then
            DFS(w)
        end if
    end for
end function
lastNr ← 0
for all v ∈ Nodes do
    visited[v] ← False
end for
for all v ∈ Nodes do                      ▷ nodes of the graph are taken in random order
    if not visited[v] then
        DFS(v)
    end if
end for
```

Let *DFS_Order* be the language of words of the form $n\$matrix\$order$ such that for some directed graph G, and for some run of the above algorithm (i.e., for some order in which nodes are considered in the "for" loops),

- n is the number of nodes of G, written in binary,
- *matrix* is a string consisting of n^2 bits, where 1 on position $n \cdot (i-1) + j$, for $1 \leq i, j \leq n$, means that there is an edge from node i to node j, and 0 that there is no such edge (there cannot exist an edge from i to i),
- *order* is a sequence of n binary numbers, separated by \$ signs, where the i-th number is the value assigned by the algorithm to $nr[i]$.

The goal is to prove that *DFS_Order* is in **L**.