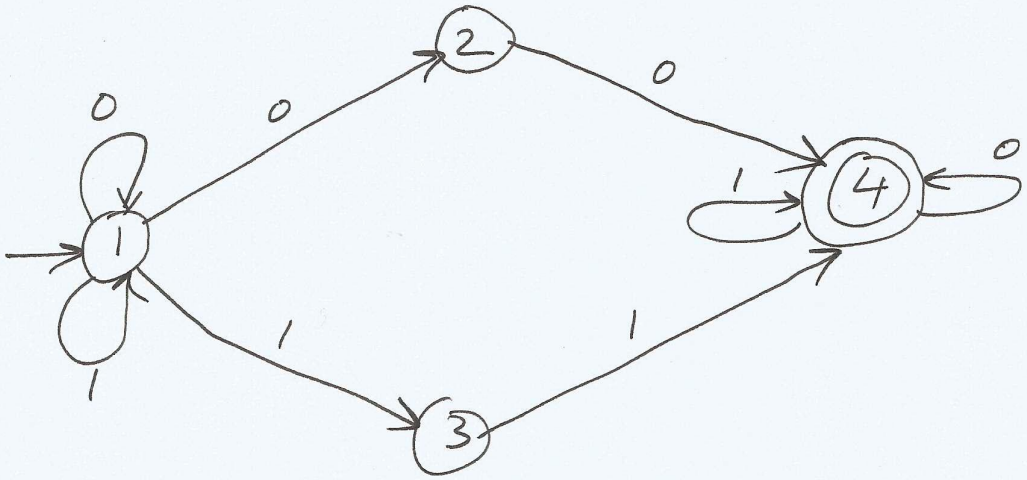
Lab Assignment 10: Simulating a Nondeterministic Finite Automata (NFA): An NFA is a directed multigraph which can have edges with self loop. Each edge is labeled with either 0 or 1. Each vertex is also labeled with either 0 or 1. Vertices represent the states of the NFA. A vertex with a label "0" represents a non-accepting state. A vertex with a label "1" represents an accepting state. Vertices are numbered in sequence starting from 1. Vertex numbered "1" will always be the start state. A start state can be either an accepting state or a non-accepting state. An input binary string is given to the NFA: $W = x_1 x_2 \ldots x_n$. (W can also be the empty string ϵ of length 0). We start with the vertex 1 which is always a start state. From vertex 1, we follow the edge labeled x_1 ($x_1 \in \{0,1\}$) to another state (vertex) q_{v_2}. Now from q_{v_2} we follow an edge labeled x_2 to q_{v_3}... and so on. Finally from state q_{v_n}, we follow an edge labeled x_n to state $q_{v_{n+1}}$. The word "nondeterministic" in NFA means that at any state q_{v_i}, we can have more than one choice of following an edge labeled x_i. The NFA consumes the input $W = x_1 \ldots x_n$ and it can be in a set of states Q' if we consider all possible moves of the NFA.

If Q' contains at least one final state, then we say that the NFA has accepted the input w. If all states in Q' are non accepting, then we say that the NFA has rejected the input w.

Example of an NFA :



Start state is shown with an arrow (vertex "1"). Accepting states are vertices with a double circle. (vertices with label 1). Vertex "4" is the only accepting state. Non-accepting states are vertices with a single circle. (Vertices with label 0). The non-accepting states are: ~~Vertex "1"~~ Vertices "1", "2", and "3".

Consider the input 1010. In state 1, we have two possibilities: either follow the self loop and remain in state 1, or follow the edge labeled 1 to go to state 3.

$\{(1)\}1010 \Rightarrow \{0, (3)\}010.$

In state 3, there is no edge labeled 0. So the computation will die out. In state 1 we have two possibilities :

Either follow the self loop to state 1 or the edge labeled 0 to state 2:

$\{①, ③\}\,0\,1\,0 \Rightarrow \{①, ②\}\,1\,0$

Now there is no edge labeled 1 from state 2. The computation will die out. From ① we have two possibilities: either follow the self loop to 1 or follow the edge labeled 1 to state 3:

$\{①, ②\}\,1\,0 \Rightarrow \{①, ③\}\,0$

In state 3, there is no edge labeled 0. So the computation will die out. In state 1 we have two possibilities: either follow the self loop to state 1 or the edge labeled 0 to state 2:

$\{①, ③\}\,0 \Rightarrow \{①, ②\}$

Now the NFA has consumed the input. It can be in either state 1 or state 2. Both are non-accepting states. So the NFA rejects the input 1010.

Now consider the input 1100:

$\{①\}\,1\,1\,0\,0 \Rightarrow \{①, ③\}\,1\,0\,0 \Rightarrow \{①, ③, ④\}\,0\,0$

$\Rightarrow \{①, ②, ④\}\,0 \Rightarrow \{①, ②, ④\}$.

Now the NFA has consumed the input. It can be in either of the states 1, 2, or 4. "4" is the accepting state. So the NFA accepts the string 1100.

You can easily verify that the given NFA accepts all binary strings with "00" or "11" (or both) as a substring.

Input: You will be given the adjacency list representation of the NFA in the following format:

Number of Vertices in the NFA

1 Label of Vertex 1 Number of outgoing edges from 1 Label of edge 1 Destination Vertex of edge 1 ...

2 Label of Vertex 2 - - -

:

For example, the given NFA is represented as:

Sample Input:

```
4
1 0 4 0 1 0 2 1 1 1 3
2 0 1 0 4
3 0 1 1 4
4 1 2 0 4 1 4
```

You have to output the first 10 binary strings in lexicographic order: {e, 0, 1, 00, 01, 10, 11, 000, 001, ...} which are accepted by the NFA. (e is the empty string).

Sample Output:

```
00
11
000
001
011
100
110
111
0000
0001
```

Procedure: You can use Backtracking for solving this problem. Please refer "Backtracking Tutorial" on Moodle.