IIT PALAKKAD

COMPUTER SCIENCE AND ENGINEERING
Indian Institute of Technology, Palakkad
CS2130: Data Structures and Algorithms Lab
Lab 7 (Depth First Search)

28 Sep, 2018

Time: 3 hrs

General Instructions

- You are expected to follow good programming practices (refer to the book titled *C How to Program* by Dietel and Dietel). Your programs should handle all kinds of test cases - positive, negative, corner cases/boundary conditions, good input, bad input and are never expected to crash on any input.
- You are also expected to provided a high level description of the functions used in your programs. There is no restriction on the format except that it should be concise and precise. Further, the running time and additional space (excluding the space for the arguments) used by the functions should be clearly stated and justified.
- **Please note that poor programming practices and improper documentation can attract a penalty of up to 20% of the marks allocated to a question.**

Each of the following programs take a directed graph G on n vertices and m edges as input where $n \geq 1$. The vertex set of G is $\{0, 1, \ldots, n-1\}$ and an edge directed from vertex i to vertex j is denoted by (i, j). The input file will contain $m+1$ lines with the first line containing the number n. Each of the subsequent m lines will contain two non-negative integers i and j, separated by a space, denoting edge (i, j) of G.

1. (30 points) Write a program to represent a directed graph G on n vertices using the adjacency list representation. The output should contain n lines denoting the adjacency list representation of G where the ith line contains the set of out-neighbours of vertex $(i-1)$. The elements of this set should be displayed in increasing order of their identifiers and each element must be followed by \$. If vertex j has no out-neighbour, then the $(j+1)$th line only contains \$.

```
Sample Input:
5
0 2
0 3
0 4
3 1
1 2
4 2

Expected Output:
2$3$4$
2$
$
```

Continued on next page...

```
1$
2$
```

2. (30 points) Write a program to implement the Depth First Search algorithm on directed graphs. At any point in the execution of the algorithm, if there are multiple choices for a vertex to be picked, then pick the one with least identifier. The output is the edges of the resulting unique DFS forest F. If F consists of ℓ edges then the output file should consists of ℓ lines with each line denoting an edge of F. Each such line should be of the form (i, j) where (i, j) is an edge in F. Further, for each $0 \leq i \leq n-1$, if F has ℓ_i edges from vertex i, then Lines $(i \cdot \ell_{i-1} + 1)$ to $(i \cdot \ell_{i-1} + \ell_i)$ of the output should be these edges displayed in the increasing order of their second endpoints. That is, if (i, k) and (i, j) are edges in F with $k > j$, then (i, j) should be displayed before (i, k).

```
Sample Input:
5
0 2
0 3
0 4
3 1
1 2
4 2

Expected Output:
(0,2)
(0,3)
(0,4)
(3,1)
```

3. (20 points) Modify the program designed in the previous question to record the start time and the finish time for each vertex during the execution of DFS. Recall that the start time of a vertex is the instant at which it is discovered (or visited) and the finish time of a vertex is the instant at which it's exploration is completed. The start times and finish times are integers between 1 and $2n$ (both inclusive) where n is the number of vertices of the input graph. The output of this program consists of n lines where the ith line is of the form x/y denoting that vertex i was discovered at time step x and its exploration was completed in time step y.

```
Sample Input:
5
0 2
0 3
0 4
3 1
1 2
4 2


Expected Output:
```

Continued on next page...

```
1/10
5/6
2/3
4/7
8/9
```

4. (20 points) Modify the program designed in the previous question to determine if an input directed graph G is acyclic or not. If it is indeed acyclic, then output the topological order of G that is obtained by arranging its vertices in the decreasing order of their finish times. The output consists of a single line containing this topological order with consecutive vertices separated by a space. If the input graph has a cycle, then the output is -1.

```
Sample Input 1:
5
0 2
0 3
0 4
3 1
1 2
4 2

Expected Output:
0 4 3 1 2

Sample Input 2:
5
0 1
0 4
1 2
2 3
3 1

Expected Output:
-1
```