Algorithms

Lab 6 (8%)

Topics: Elementary Graph Algorithms;
Textbook: Cormen Introduction to Algorithms 3rd ed.

Problem 01 (2%)
Depth-First Search Algorithm (DFS)

1. Read Cormen 22.1, 22.3, Appendix B.4
2. Use this algorithm to implement following C++ program:

Given a number of vertices of undirected unweighted graph and adjacency matrix describing this graph. Program has to output the number of connected components in this graph. Use adjacency lists to represent graph in your program.

Sample Input:
7
0001000
0010110
0100000
1000000
0100000
0100001
0000010

Sample Output:
2

Problem 02 (2%)
Topological Sorting

1. Read Cormen 22.1, 22.3, 22.4, Appendix B.4
2. Use this algorithm to implement following C++ program:

Given a directed unweighted acyclic graph (DAG) represented by a number of vertices and following by n adjacency lists (i + 1 line in input represents adjacency list for vertex i). Given graph represents a system of prerequisites of university courses. Your program has to output sequence of all courses representing the order which students have to follow in their educational tracks without violation of prerequisites' system. If given graph is not a DAG, print message "loop".

Sample Input:
```
11


8 10

11

11 8
9


2 9 10
```

Note:
Empty lines represent vertices which are not adjacent to any other vertices

Sample Output:
```
res: 7 6 5 11 4 3 10 8 9 2 1
```

Problem 03 (2%)
Breadth-First Search Algorithm (BFS)

1. Read Cormen 22.1, 22.2, Appendix B.4
2. Use this algorithm to implement following C++ program:

Given a number of edges of undirected unweighted graph, a list of graph's edges (each line represents one edge, vertices are represented by arbitrary strings without whitespaces) , source and destination vertices . Program has to output shortest distances to all vertices of the graph from source vertex and a a shortest path from source vertex to destination vertex as a sequence of vertices. Use adjacency lists to represent graph in your program.

Sample Input:
```
10
r s
t u
r v
s w
t w
t x
u x
u y
w x
x y
s
y
```

Notes:
s is a source vertex
y is a destination vertex

Sample Output:
```
r: 1
s: 0
t: 2
u: 3
v: 2
w: 1
x: 2
y: 3
 s w x y
```

Problem 04 (2%)

BFS. Shortest distance in a maze

Use BFS algorithm to find shortest distance in a maze from given source point to given destination point.

Sample Input:

```
10
S#...#..#D
.#...#..#.
..........
.#...#..#.
.#...#..#.
.#...#..#.
.#...#..#.
.....#..#.
.#...#..#.
.#...#....
```

Notes:

10 is a size of maze

`#` is a wall

`.` is an empty space

`S` is a source point

`D` is destination point

Sample Output:

```
S#...#..#*
*#...#..#*
**********
.#...#..#.
.#...#..#.
.#...#..#.
.#...#..#.
.....#..#.
.#...#..#.
.#...#....
```