Institute of Engineering & Management
Department of Computer Science & Engineering
Design & Analysis of Algorithm Lab for 3rd year 5th semester 2018
Code: CS 591

Date: 3/10/18

WEEK-5

Assignment-1

Problem Statement: Given a directed acyclic graph, write an algorithm for finding the depth.

Algorithm:

Name: Ranajit Roy, **Sec:** A, **Roll:** 47

Source code:

```
#include <iostream>
#include <vector>
#include <list>
#include <queue>

int main()
{
  std::cout<<"Enter the no. of vertices & edges: ";
  int v, e; std::cin>>v>>e;
  std::vector<std::list<int>> adjList(v);
  std::cout<<"Enter the edges with adjacent vertices:\n";
  for(int i = 0; i < e; i++)
  {
        int x, y; std::cin>>x>>y;
        adjList[x].push_back(y);
  }
  std::vector<bool> visited(v,false);
  std::vector<int> depth(v,0);
  std::cout<<"Enter the source: ";
  int start; std::cin>>start;
  std::queue<int> q;
  q.push(start);
  while(!q.empty())
  {
        int x = q.front();
        q.pop();
        visited[x] = true;
        for(auto &i: adjList[x])
        {
              if(!visited[i])
              {
                    q.push(i);
                    visited[i] = true;
                    depth[i] = depth[x] + 1;
              }
        }
  }
  std::cout<<"Depth for each vertex:\n";
  for(auto i=0;i<v;i++)
  {
        std::cout<<i<<" -> "<<depth[i]<<"\n";
  }
}
```

Screen-Shot:

```
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ g++ bfs.cpp
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ ./a.out
Enter the no. of vertices & edges: 7 8
Enter the edges with adjacent vertices:
0 1
0 2
1 2
2 6
2 3
3 4
3 5
4 5
Enter the source: 0
Depth for each vertex:
0 -> 0
1 -> 1
2 -> 1
3 -> 2
4 -> 3
5 -> 3
6 -> 2
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$
```

Time Complexity:

Source code:

```
#include <iostream>
#include <queue>
#include <vector>
#include <list>
#include <utility>
#include <algorithm>

struct compare
{
  bool operator()(const std::pair<int,int> x, const std::pair<int,int y)
  {
        return x.second >= y.second;
  }
};

int main()
{
  std::cout<<"Enter the no. of vertices & edges: ";
  int e, v; std::cin>>v>>e;
  std::vector<std::list<std::pair<int, int>>> adjList(v);
  int inf = 1;
  std::cout<<"Enter the edges with adjacent vertices and their
                                                        weights:\n";
  for(int i = 0; i < e; i++)
  {
        int x, y, z; std::cin>>x>>y>>z;
        adjList[x].push_back(std::make_pair(y, z));
        inf += z;
  }
  std::vector<int> distance(v, inf);
  std::priority_queue<std::pair<int, int>, std::vector<std::pair<int,
                                                  int>>, compare> pq;
  std::cout<<"Enter the source: ";
  int src; std::cin>>src;
  distance[src] = 0;
  pq.push(std::make_pair(src, 0));
  while(!pq.empty())
  {
        int temp = pq.top().first;
        pq.pop();
        for(auto& i : adjList[temp])
        {
              int v = i.first;
              if(distance[v] > distance[temp] + i.second)
              {
                    distance[v] = distance[temp] + i.second;
                    pq.push(std::make_pair(v, distance[v]));
              }
        }
  }
  std::cout<<"Shortest distances:\n";
  for(int i = 0; i < v; i++)
        std::cout<<src<<"->"<<i<<" = "<<distance[i]<<'\n';
}
```

Screen-Shot:

```
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ g++ dik.cc
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ ./a.out
Enter the no. of vertices & edges: 7 8
Enter the edges with adjacent vertices and their weights:
0 1 1
0 2 5
1 2 2
2 6 1
2 3 3
3 4 2
3 5 2
4 5 1
Enter the source: 0
Shortest distances:
0->0 = 0
0->1 = 1
0->2 = 3
0->3 = 6
0->4 = 8
0->5 = 8
0->6 = 4
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$
```

Time Complexity:

Source code:

```
#include <iostream>
#include <queue>
#include <vector>
#include <list>
#include <utility>
#include <algorithm>

int main()
{
  std::cout<<"Enter the no. of vertices & edges: ";
  int e, v; std::cin>>v>>e;
  std::vector<std::list<std::pair<int, int>>> adjList(v);
  int inf = 1;
  std::cout<<"Enter the edges with adjacent vertices and their
                                                  weights:\n";
  for(int i = 0; i < e; i++)
  {
        int x, y, z; std::cin>>x>>y>>z;
        adjList[x].push_back(std::make_pair(y, z));
        inf += z;
  }
  std::vector<int> distance(v, inf);
  std::cout<<"Enter the source: ";
  int src; std::cin>>src;
  distance[src] = 0;
  for(int i = 0;i < v-1; i++)
  {
        std::vector<int> temp(distance);
        for(auto j=0;j<v;j++)
        {
            for(auto k = adjList[j].begin();k!=adjList[j].end();k++)
            {
                if(distance[k->first] > distance[j]+k->second)
                {
                    temp[k->first] = distance[j] + k->second;
                }
            }
        }
        distance = temp;
  }
  std::cout<<"Shortest distances:\n";
  for(int i = 0; i < v; i++)
        std::cout<<src<<"->"<<i<<" = "<<distance[i]<<'\n';
}
```

Screen-Shot:

```
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ g++ belf.cc
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ ./a.out
Enter the no. of vertices & edges: 7 8
Enter the edges with adjacent vertices and their weights:
0 1 1
0 2 -5
1 2 2
2 6 1
2 3 3
3 4 2
3 5 -2
4 5 1
Enter the source: 0
Shortest distances:
0->0 = 0
0->1 = 1
0->2 = -5
0->3 = -2
0->4 = 0
0->5 = -4
0->6 = -4
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$
```

Time Complexity:

Source code:

```
#include <iostream>
#include <cmath>

int main()
{
  int s, d, smd=0;
  std::cout<<"Enter the sum and no. of digits: ";
  std::cin>>s>>d;
  if((double) s/d > 9)
  {
        std::cout<<"No number possible!!\n";
        return 1;
  }
  for(int i = 0; i < d; i++)
  {
        if(i == d-1)
        {
              smd = s*(int)pow(10, (double)i) + smd;
        }
        else if(s>9)
        {
              smd = 9*(int)pow(10, (double)i) + smd;
              s -= 9;
        }
        else{
              smd = (s-1)*(int)pow(10, (double)i) + smd;
              s = 1;
        }
  }
  std::cout<<"The minimum number possible: "<<smd<<"\n";
}
```

Screen-Shot:

```
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ g++ dig.cc
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$ ./a.out
Enter the sum and no. of digits: 18 4
The minimum number possible: 1089
rana@ranajit:~/Git/College_programs/5th SEM/Algorithm/Week 5$
```

Time Complexity: