CSCE 1101 Spring 2023: Fundamentals of Computing II
Assignment #3

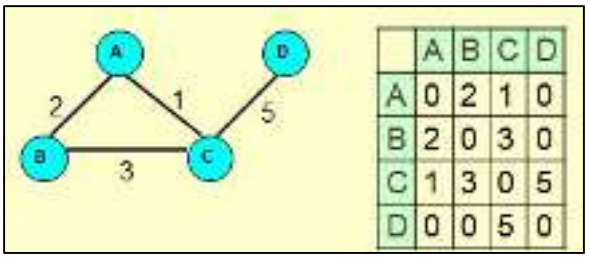
Dr. Amr Goneid ***Date: Thu March 9, Due: Sun March 19, 2023***

In the following programming assignment, you need to be familiar with Template classes and the Stack ADT.

Graph Traversal

A network of motorways connects N cities. Such network can be represented by a graph with N nodes where each node represents a city and the weight of an edge represents the distance of the motorway between a pairs of cities.

Such graph can be represented in memory using an Adjacency Matrix. If the number of vertices (cities) is N, this matrix is a 2-D array with N rows and N columns labeled with the cities names. An entry in the matrix at row (i) and column (j), say a(i,j) represents the weight (distance) of the edge (motorway) between city (i) and city (j) (see example on the right).



	A	B	C	D
A	0	2	1	0
B	2	0	3	0
C	1	3	0	5
D	0	0	5	0

If an entry a(i,j) ≠ 0 then vertices (i) and (j) are adjacent, i.e., there is an edge between the two vertices. Setting a(i,j) = 0 is usually used to indicate the absence of an edge or the distance between an edge and itself. Also, the graph is called **Undirected** if the edge allows both directions between vertices (i) and (j). For such undirected **weighted** graphs, a(i,j) = a(j,i), and the adjacency matrix is symmetric. The example shown represents N = 4 cities connected by undirected motorways.

Objective:

From a given source city (say A), traverse the graph (visit every other city in the network) to find the order of city visits and compute the total distance traversed.

Graph Traversal Algorithm

One method to traverse a graph is to use the ***DFS*** (Depth-First Search) algorithm. A non-recursive version uses a *stack* ADT:
Let the indices of the cities be 1 , 2, ..N and **k** be the index of the source city,

A[1..N, 1..N] = a 2-D array representing the adjacency matrix of the graph where A_{ij} is the distance between city (i) and city (j),
V[1..N] = an array recording the order (1,2, …) of visits, where V_i is finally the order of visiting city (i).
S = a stack of integers
unseen, hold = current state of a node

```
Algorithm DFS (k)
// k is the index of the start city
// Assume "hold" to indicate a node discovered but not yet visited
{
  Initialize V array to all unseen;
  Set order = 0;
  Create an Empty stack S;
  Push (k) on the stack (S);
  while stack (S) is not empty
    {
          Pop stack (S) to get current source (i);
          Increment (order) and save it in V[i];
          Scan row (i) in adjacency matrix from right to left (j = N downto 1)
            if (city (j) is connected to city (i)):
              if (V[j] == unseen) { Push (j) on (S);  Set V[j] = hold; }
    }
}
```

Input Data:

The zip file **"Cities.rar"** contains an Excel sheet file **"Cities15.xlsx"** representing the adjacency matrix of a weighted graph for N = 15 cities. The cities are simply named (A, B, C,….). The distance between city (i) and city (j) is given in the sheet at row (i) and column (j). The graph is connected (there is a path from every city to every other city) and the distances are all positive integers in the range 32 – 98. Zero distance represents the absence of a motorway, or the distance between a city and itself.

Required Implementations:

1. Implement an array-based stack template class ***Stackt.***
2. Save the given graph to a text file.
3. Develop a program to input the graph file and to use the algorithm given above to traverse the graph (visit every other city in the network) from a given source city to find the order of city visits. **Allow for graphs of variable sizes N.**
4. Compute and report the total distance traversed.