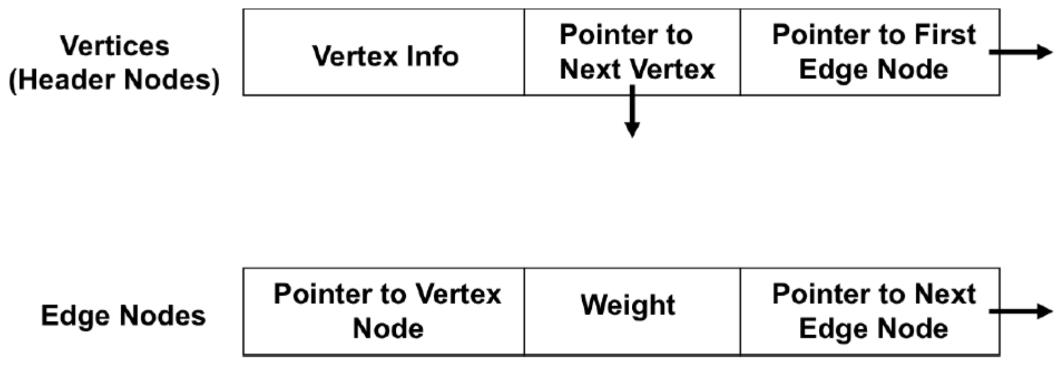
Homework 01 – Graph

You will implement a Graph ADT class. This class uses adjacency list as the internal representation of the graph structure. You will implement DFS and BFS algorithms using the adjacency list representation of the graph. Note that *stack* and *queue* containers from the Standard Template Library will be used for DFS and BFS implementation. A sample BFS implementation is given below.

Main.cpp is a driver program to test Graph classes. graph.h is given. Your task is to complete the graph.cpp file that implements the member function of Graph class. Input files are given. You can use the given makefile to compile this program.

VertexNode and EdgeNode are structs that are described in graph.h file. They are shown in the figure below:



```
//BFS algorithm implementation:

void Graph::BreadthFirstSearch(string startVertex, string endVertex, queue<string>& visitedq)
{
   queue<string> q;
   queue<string> adjQ;

   if ( (VertexExists(startVertex) == NULL) || (VertexExists(endVertex) == NULL) )
        throw GraphVertexNotFound();

   bool found = false;
   string vertex;
   string item;

   ClearMarks();
   q.push(startVertex);
   do
```

```
    {
       vertex = q.front();
       q.pop();
       if (vertex == endVertex)
       {
          found = true;
          visitedq.push(vertex);
       }
      else
      {
         if (!IsMarked(vertex))
         {
            MarkVertex(vertex);
            visitedq.push(vertex);
            GetToVertices(vertex, adjQ);

            while (! adjQ.empty() )
            {
               item = adjQ.front();
           adjQ.pop();

           if (!IsMarked(item))
              q.push(item);
            }
         }
      }
    } while (!q.empty() && !found);

    if (!found)
    {
         while (!visitedq.empty())
         {
          visitedq.pop();
         }
    }

} // End BreadthFirstSearch()
```