Middle East Technical University | Department of Computer Engineering

CENG 242

Programming Language Concepts

Spring '2012-2013

Programming Assignment 4

Due date: 15 May 2013, Wednesday, 23:55

1 Introduction

Remember the database we hacked before, now we realized that we actually need the implementation in C++.
Header files are already written and we thought you can help us again.

2 Specifications

As the header files are supplied. You should implement .cpp files (Node.cpp, Edge.cpp, Network.cpp). You must not change header files.

1. **Node**
   A node has a label and a list of edges (adjacencyList) that contains "Edge"s. You should implement

   - **Constructor**
   - **Destructor**
   - **get-set** methods for label to read and change label
   - **addEdge** to add an edge from a node to another node. You are going to use Edge class for that. When you are trying to add a connection from NodeA ->NodeB if there is already an edge (with same or different weight) you should throw an exception EDGE_ALREADY_EXISTS.
   - **removeEdge** to remove an edge when you are given a name of the adjacent. If there is no adjacent node with the given name you should throw EDGE_DOES_NOT_EXIST.
   - **operator+** You should override + operator in order to merge connections of a node. You will need to merge two networks and when doing so, you should merge the connections of node. That's why we need this overloading.
   - **operator==** You should override == operator in order to compare if two Nodes are the same.
   - **operator<<** You should override <<operator in order to be able to print out a node. For example if you are printing a node named A with outgoing connections to B and C nodes, the output should be as below.

     ```
     A [(B,5),(C,4)]
     ```

- **Edge**
  An edge is a connection between two nodes. As it can be added to a node, it should be showing a direction to another node and it has a weight. You should implement
  - **Constructor**
  - **Destructor**
  - **get** method for weight
  - **get** method for adjacent node
  - **operator==** to compare if two edges are the same
  - **operator<<** to output an edge.
- **Network**
  - **Constructor**
  - **Destructor**
  - **addNode** to add a node to the network with the given name. You should throw NODE_ALREADY_EXISTS if there is a node with the same name in the graph.
  - **removeNode** to remove a node from the network. You should throw NODE_DOES_NOT_EXIST if there is not a node with the given name in the network.
  - **addEdge** to add an edge with the given weight to the network between the nodes with the given name. You should throw SOURCE_NODE_DOES_NOT_EXIST if there is no source nodes with the given name and you should throw DESTINATION_NODE_DOES_NOT_EXIST if there is no such destination node.
  - **removeEdge** to remove an edge from the network between the nodes with the given name. You should throw SOURCE_NODE_DOES_NOT_EXIST if there is no source nodes with the given name and you should throw DESTINATION_NODE_DOES_NOT_EXIST if there is no such destination node. If there is both the source and destination nodes but no edge from source to destination you should throw EDGE_DOES_NOT_EXIST
  - **operator+** You should override + operator in order to merge two networks. Networks "can" have no common edges this time to ease your job.
  - **operator==** You should override == operator in order to compare if two Networks are the same.
  - **operator<<** You should override <<operator in order to be able to print out whole Network.

3 Regulations

1. **Programming Language: C++**
2. **You will be given header files and a makefile.**
3. **Cheating: We have zero tolerance policy for cheating**. People involved in cheating will be punished according to the university regulations.
4. **Newsgroup:** You must follow the newsgroup (news.ceng.metu.edu.tr) for discussions and possible updates on a daily basis.
5. **Evaluation:** Your program will be evaluated automatically using "black-box" technique so make sure to obey the specifications.

4 Submission

Submission will be done via COW. Create a tar.gz file named hw4.tar.gz that contains all your cpp files (Node.cpp Edge.cpp Network.cpp)
Note: The submitted archive should not contain any directories!