**Documentation**

**Subject: Data Structures (CS)**

**Project Name: The Network Emulator**

**Group Members:**

Tayyab Meer (i16-0171)

Maryam Bokhari (i16-0059)

Rehan Imran (i16-0116)

**Classes we have created:**

* **B-trees**
* **B-tree Node**
* **S-tree**
* **List**
* **Router**
* **Queue**
* **Array**
* **Heap**
* **Graph**
* **Routing Table**
* **RT**
* **Message**

**Graph Loading:**

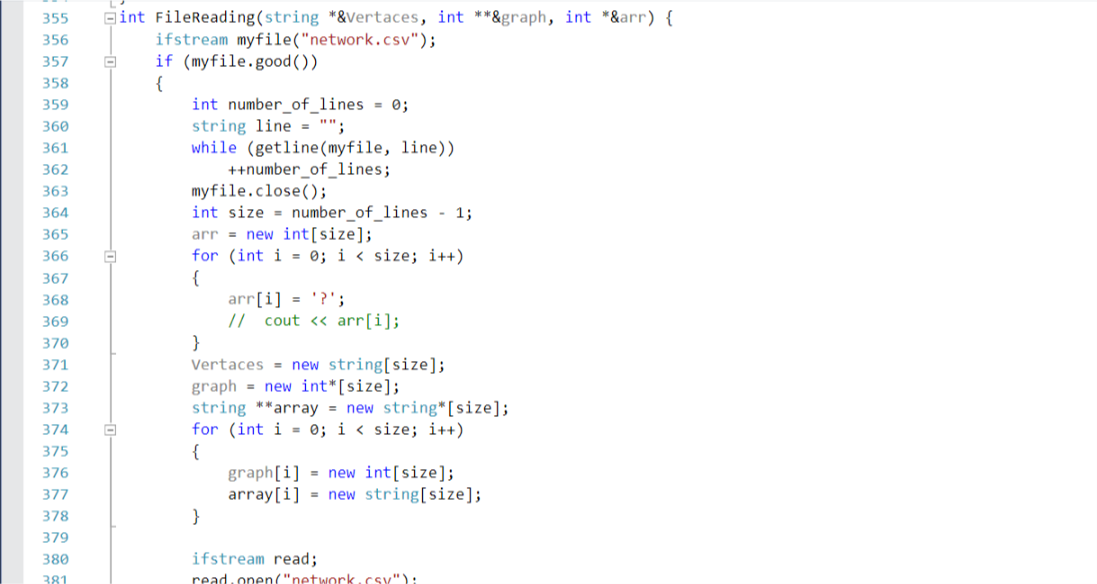
We are reading a .csv file through fstream class in C++. We read the first line of the graph and we sorted it into monitors and routers. The question marks in the file represents that there is no connection between monitor to monitor, monitor to router, and router to router. Monitors’ edge carries its priority while the router’s edge represents the weightage.

Figure 1: File Reading Function

**Creation of Routing Tables using Dijkstra:**

Dijkastra algorithm takes a source and a destination and returns the shortest path between the nodes. Here we took each machine and router and ran it across all the machines. We created the routing table which included two string elements, destination and next best.

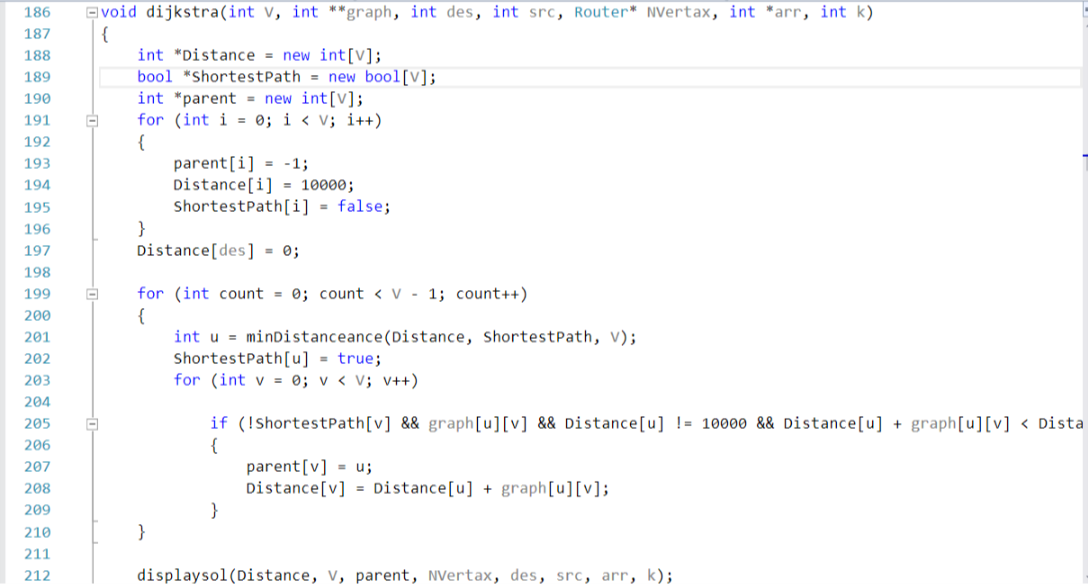


Figure 2: Dijkstra algorithm function

**Dynamically changing of Graph:**

* **Delete Router:**

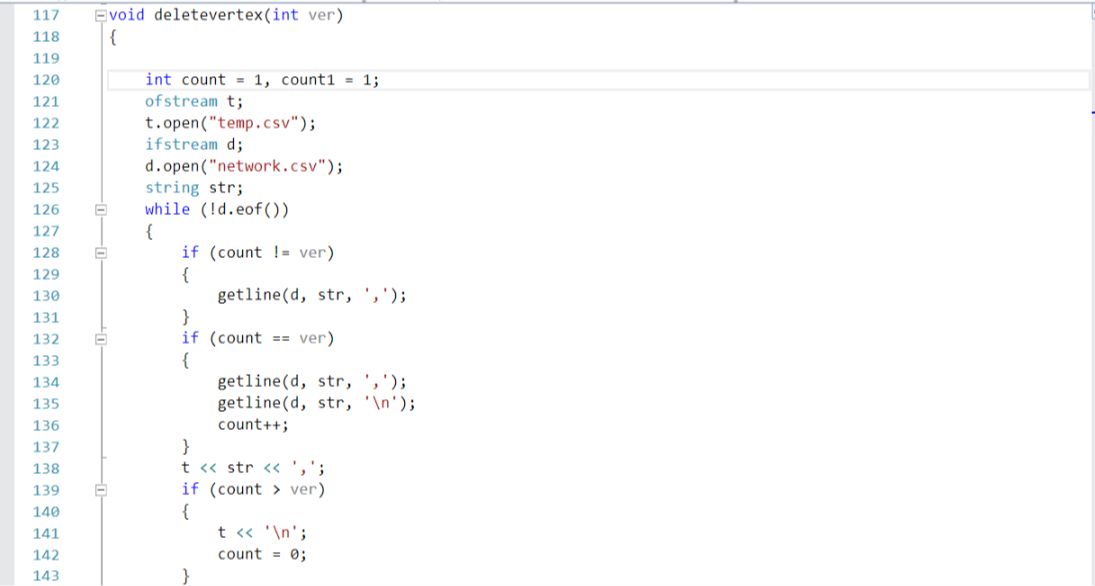
****

Figure 3: Delete Router function

**Priority base Routing of messages:**

There is a message class. In main messages are read through a file or can be input one by one. Each router has a binary heap which inputs all the messages coming from the outgoing of another router or is directly sent from a monitor. Data structures used in it are Array and Heap. The incoming queue of each router is the part of RT.

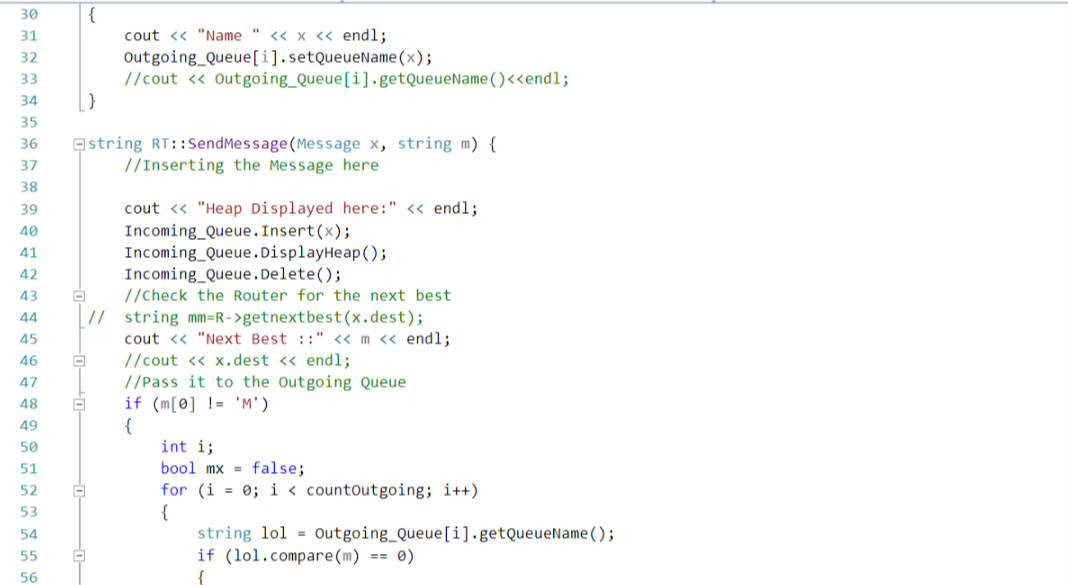


Figure 4: Sending messages in incoming queue

**Simulation of Latency:**

RT has a function SendMessage(). In this function ,sleep is called for one second during the transaction of message from a router’s outgoing queue to another router’s incoming queue.



Figure 5: Delay function

**Functionality of different data structures:**

1. **Array:**

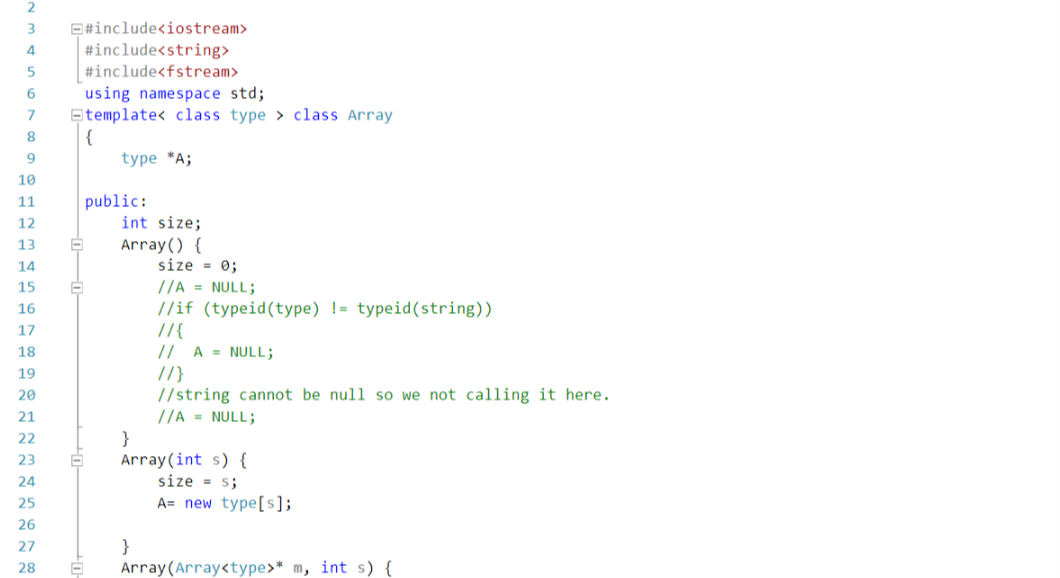
A template based array implementation that dynamically grows and reduces. In this emulator we are making the array of messages, which we are using in Heap and Queue data structures.

Figure 6: Array Implementation

1. **B-trees:**

A data structure which gets populated by dijkstra algorithm. The B-tree Node contains the destinations and the next best routers for each monitor and router. It has a root pointer to a b-tree node and the minimum degree of a b-tree.

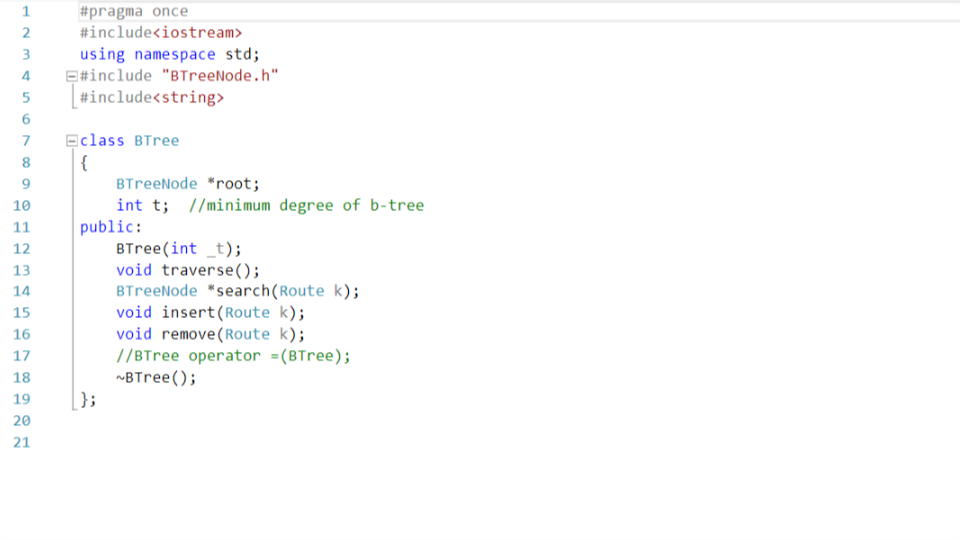


Figure 7: Btree implementation

1. **Splay trees:**

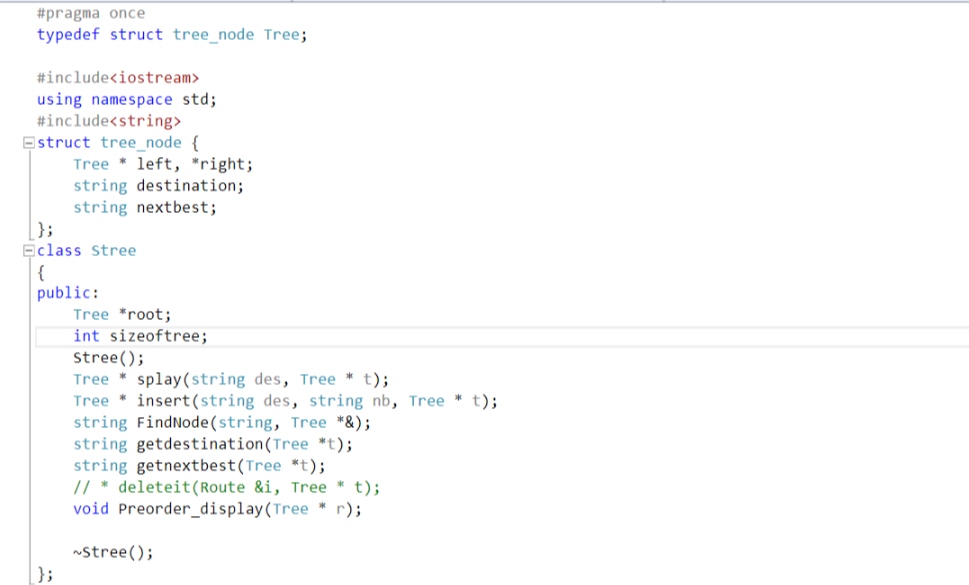
Populated by dijkstra algorithm and contains the destinations and the next best routers from source monitor. Splay tree node has its item as well as left and right pointers. Splay tree have a root pointer to splay tree node. After each call to Splay(), the destination that has been accessed becomes the root of the tree.

Figure 8: Stree Implementation

1. **Linked List:**

Data Structure that is again populated by Dijkastra. The number of steps to access the next best router is 0(n). The node of linked list is named routing table. 

Figure 9: Linked List Implementation

1. **Router:**

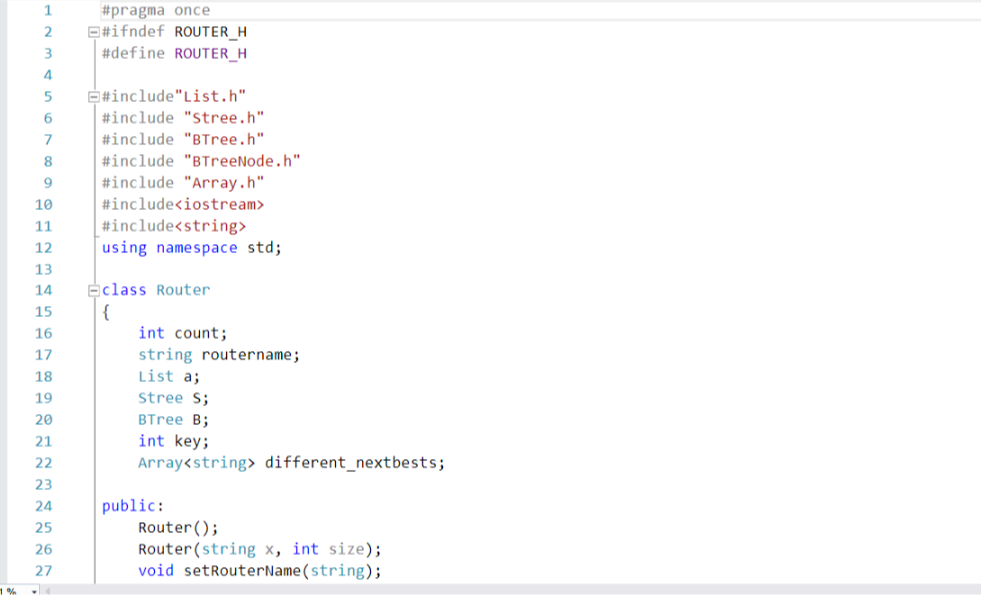
Router is the collection of the different data structures. The menu asks which data structure you want to use for implementation of the routing table. Data is entered through function populatenode(). Key is set to differentiation of which data structure to be used.

Figure 10: Router Implementation

1. **Queue:**

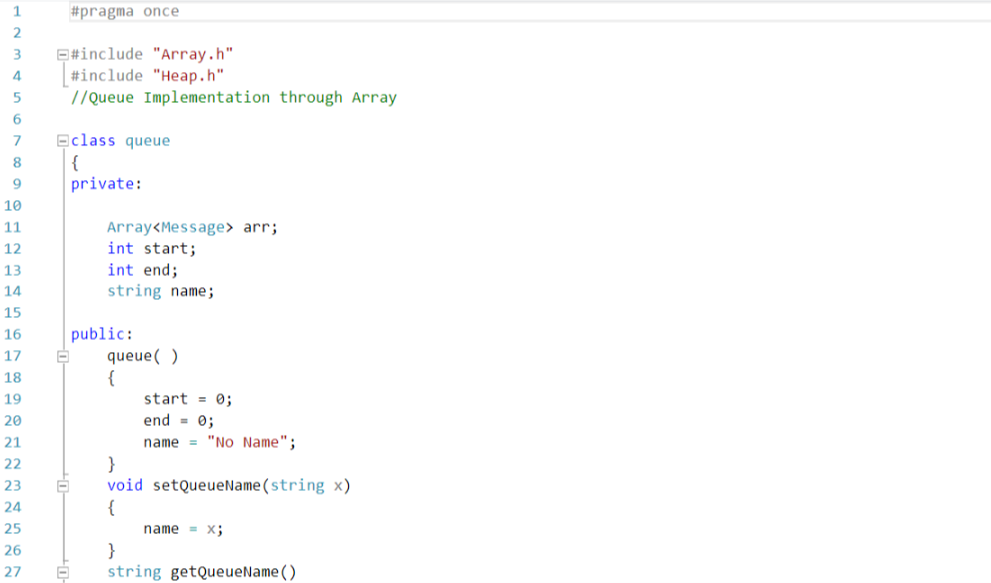
Queue is a data structure used to input messaged in First IN First Out functionality. Multiple Outgoing Queues are created in RT. 

Figure 11: Queue Implementation

1. **Heap:**

Binary Heap takes in messages and sorts them based on the priority of the messages and then processes them.

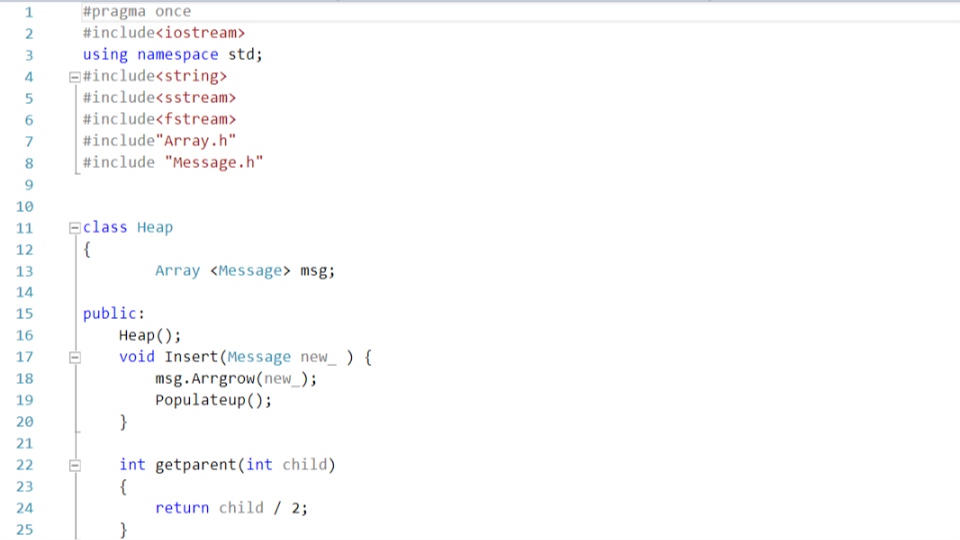


Figure 12: Heap Implementation

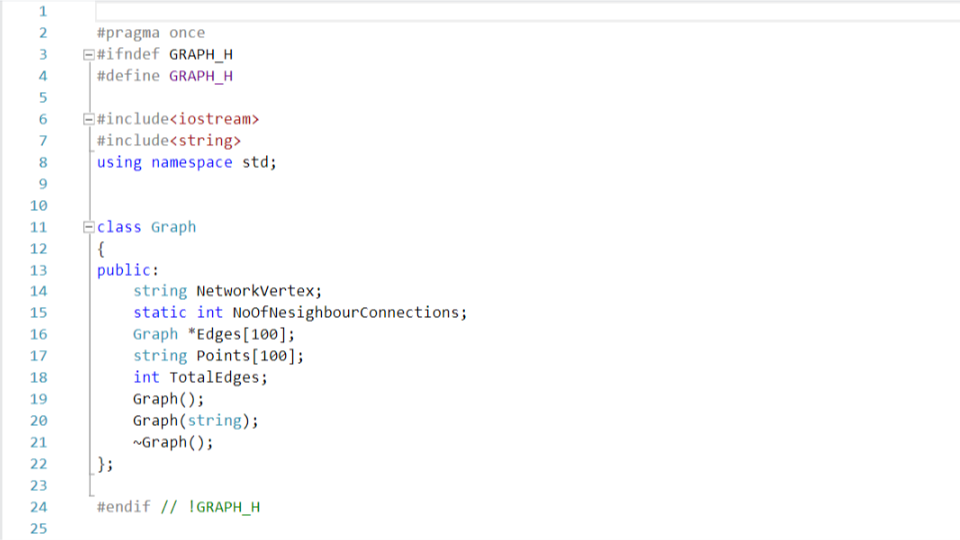
1. **Graph: **

Figure 13: Graph Implementation

1. **RT:**

Rt is a structure that is basically processing all the messages. Rt is at the root of the hierarchy. The incoming queue, the router and the outgoing queue are being populated from main in RT.

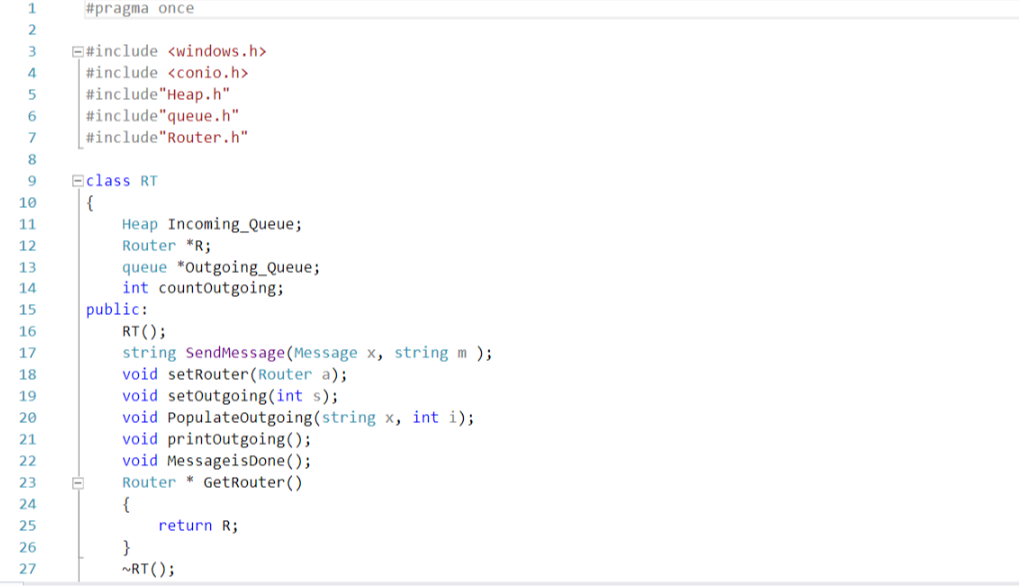
****

Figure 14: RT Implementation

**Hierarchy of the Network Emulator:**

Figure 15 : The Hierarchy of our Network Emulator