İTU COMPUTER ENGINEERING DEPARTMENT

BLG 223E DATA STRUCTURES

HOMEWORK - 3

Due Date: 22 December, 23:59

Submission:

1. Make sure you write your name and number in the source code files of your project, in the following format:

   /* @Author
   Student Name: <student_name>
   Student ID: <student_id>
   Date: <date> */

1. Use comments wherever necessary in your code to explain what you did.
2. You are not allowed to include any STL container.
3. Your program should compile and run on Linux environment using g++ (version 4.8.5 or later). You can test your program on ITU's Linux Server using SSH protocol.
4. To compile the code, you can use the following command:

   `g++ main.cpp –o main` or `g++ -std=c++11 main.cpp -o main`

5. You can execute your program by using the following command:

   `./main input1 input2`

6. After you make sure that your code is compiled smoothly, prepare a single code file named main.cpp. If you have multiple source codes, gather them together in one single cpp file.
7. Submit this file through www.ninova.itu.edu.tr. Ninova enables you to change your submission before the submission deadline.

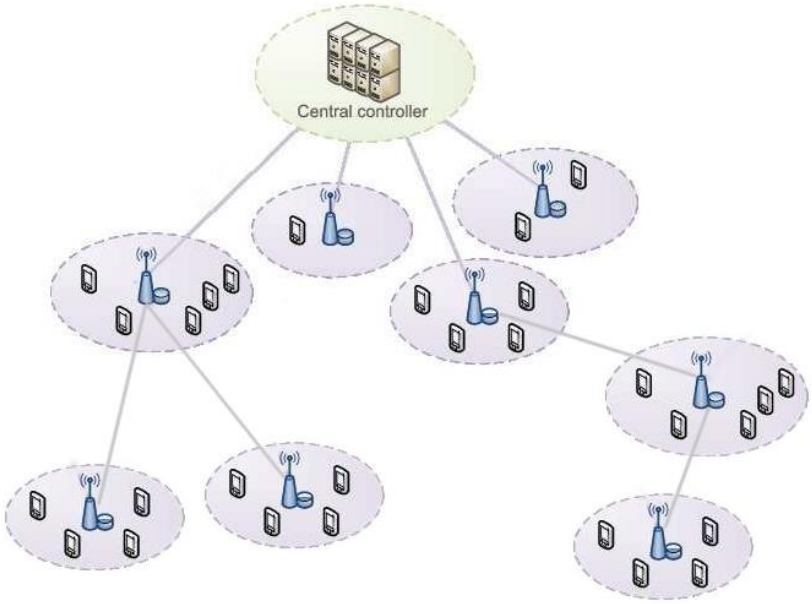
- Do not miss submission deadline. Do not leave your submission until the last minute. The submission system tends to become less responsive due to high network traffic.
- Homeworks sent via e-mail will not be graded.
- Academic dishonesty including but not limited to cheating, plagiarism and collaboration is unacceptable and subject to disciplinary actions. Your homeworks will be checked with a plagiarism checker system, any student found guilty will receive 0 as his/her grade for the homework and subject to disciplinary actions.

If you have any question about the homework, contact the teaching assistant Ismail Bilgen via Message Board in Ninova, e-mail (ibilgen@itu.edu.tr) or in 5111.

Deliver the Messages

Background: A telecommunication company aims to deliver messages to customers' mobile phones. They built a flexible architecture consisting of a Central Controller (CC), Base Stations (BSs) and Mobile Hosts (MHs) as in the figure[1]. Since MHs can move from one BS to another, the system should be designed to find the target MH first to deliver a message.

Objective: Deliver each message waiting in the delivery queue to their target MH.



A sample network[1].

Constraints:

- A message received to the central controller can only be delivered via the connected base stations to the mobile hosts.
- The central controller doesn't know the location of MH. First, it must search for the target MH. Once it finds, it can deliver the message using the path found.

Requirements: Develop a program in order to carry out the following tasks:

- Process a network file to create the network
- Process a messages file to determine each message and its target.
- Traverse recursively in deep-first manner to find the target MH and obtain path from Central Controller to the target MH.
- Print out:
    - Each visited base station nodes while traversing (don't print mobile hosts)
    - The message and the full path of the target MH if MH is found in the network
    - The warning message if MH is not found in the network.

[1]Utilized from the source for creating the figure: https://www.researchgate.net/figure/The-architecture-of-a-backhaul-limited-caching-network-BSs-are-connected-to-the-central_fig1_301843145

Input: Two text files are provided via command lines as follows:

```
./executable network_file_name messages_file_name
```

- Network.txt : The file name containing the whole network
- Messages.txt : The file name containing the list of messages and their targets

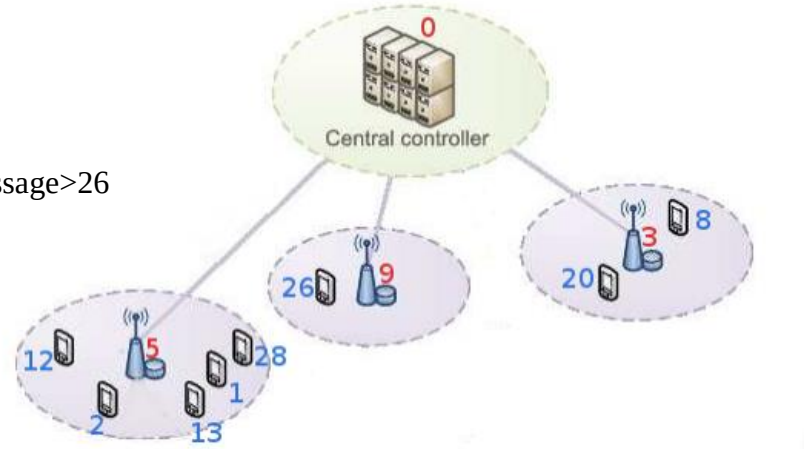
The whole network will be provided in a text file as in the example. Each row in the input file describes the type of the node, ID and parent node ID. Node type can be Base Station (BS) or Mobile Host (MH). IDs of BS and MH are unique in their own type of node groups, ie, there can be a MH and a BS having same ID number but no two BSs can have the same ID number. Parent node ID indicates the Base Station of which the node is connected to.

Assume that your program is executed as "./executable Network.txt Messages.txt" with the given Network.txt and Messages.txt files below. Network.txt corresponds to the network below.

Network.txt

BS 5 0
BS 9 0
BS 3 0
MH 12 5
MH 2 5
MH 13 5
MH 1 5
MH 28 5
MH 26 9
MH 20 3
MH 8 3

Messages.txt
---
message1>13
message2>8
This is another message>26



Read the Network.txt file line by line and create the network using a **tree structure**. Note that, the order is important. You must create the nodes of the network in the same order with the input file. Mobile hosts or base stations connected to the base station must be inserted from left to right. For example,
BSs connected to Central Controller is 5 → 9 → 3
MHs connected to BS:5 is 12 → 2 → 13 → 1 → 28

Read the Message.txt file and to get the message and its target. The message and the id of the target MH is separated by ">" sign.

Output: You must give the output in exactly the same shape with the one described below. For each message in the delivery queue,

- Print out each BS you visited while searching for the target cell.

Traversing:0 [bs_id(s)]
Ex: Traversing:0 9 2 5

➔ If the target MH is found, print out the message and full path from CC to target MH

Message:[message read from the file] To:0 [bs_id(s)] mh_[k]
Ex: Message:bla bla bla To:0 1 5 7 mh_3

➔ If the target MH not found, print out the following warning message

Can not be reached the mobile host [mh_id] at the moment

You will find a detailed example below.

Implementation Tips:

- The network will remain same as given in the Network.txt. No new MH os BS will join or leave. The connections will not change too.
- You can assume Central Controller as a Base Station with no parental node. Its ID will always be 0. It will not given in the input file. You should create it. It is connected to some BSs but not MHs.
- BS can have arbitrary number of connections, do not limit it.
- The output message must be exactly the same as described. Otherwise it can not be detected as correct since it will be compared with the answer text.
- You need to use General Tree structure (tree having arbitrary number children) for creating the network.
- You can use a queue structure for messages read from the Messages.txt since the first message (on top line of the file) will be processed first.

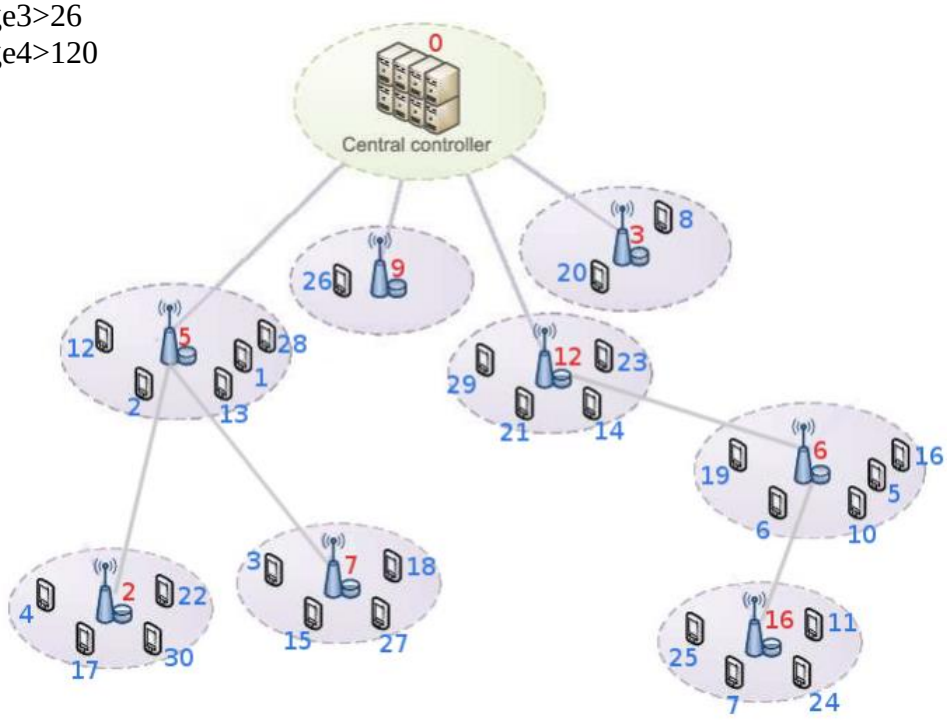
Example:

Network.txt

```
---------------
BS 5 0
BS 9 0
BS 12 0
BS 3 0
BS 2 5
BS 7 5
BS 6 12
BS 16 6
MH 12 5
MH 2 5
MH 13 5
MH 1 5
MH 28 5
MH 26 9
MH 20 3
MH 8 3
MH 4 2
MH 17 2
MH 30 2
MH 22 2
MH 3 7
MH 15 7
MH 27 7
MH 18 7
MH 29 12
MH 21 12
MH 14 12
MH 23 12
MH 19 6
MH 6 6
MH 10 6
MH 5 6
MH 16 6
MH 25 16
MH 7 16
MH 24 16
MH 11 16
```

Messages.txt

```
---------------
this is a message>13
this is another message>8
message3>26
message4>120
```



Output:

```
---------------
Traversing:0 5
Message:this is a message To:0 5 mh_13
Traversing:0 5 2 7 9 12 6 16 3
Message:this is another message To:0 3 mh_8
Traversing:0 5 2 7 9
Message:message3 To:0 9 mh_26
Traversing:0 5 2 7 9 12 6 16 3
Can not be reached the mobile host mh_120 at the moment
```