EE441- Programming Assignment 3

Due Date: 9.01.2024, 23:55

For your questions: Utkucan Doğan – utkucan@metu.edu.tr

This assignment consists of one part. You are going to create one CodeBlocks project. Don't forget to write comments to your code as they are also graded.

Ad-Hoc Network Simulation

In this homework, you are going to simulate an Ad-Hoc Network. Your simulation will have two layers with acknowledgment and resending. A header file is provided for your use. **DO NOT CHANGE ANYTHING IN THE HEADER FILE INCLUDING ITS NAME OTHERWISE YOUR HOMEWORK WILL NOT BE GRADED.**

Packet Class

1. Write a hash function for the packet class. Its calculation should include `sourceId`, `destId` and `message`.
2. Implement the constructors for the packet class. Constructors should calculate the hash value and assign it to the member parameter.
3. Implement the functions `getSourceId`, `getDestId`, `getMessage`.
4. Implement the function `checkIntegrity`. It should recalculate the hash value and compare it to the member parameter. It should return false if the values do not match.
5. Implement the function `corrupt`. It should change a random character in the `message` to a random ASCII character.

MacPacket Class

1. Implement the constructors. The default constructor should assign `MacPacketType::Empty` as type.
2. Implement the static functions which are shorthands for creating certain types of mac packets.
3. Implement the functions `getType`, `getMacSourceId`, `getMacDestId`, `getPath`, `getPacket`.

Node Class

1. Implement the constructors. The default constructor should assign the id value as zero. Throughout this project, node id zero will be considered as the invalid id/empty value.
2. Implement the functions `getId` and `getNeighbors`.

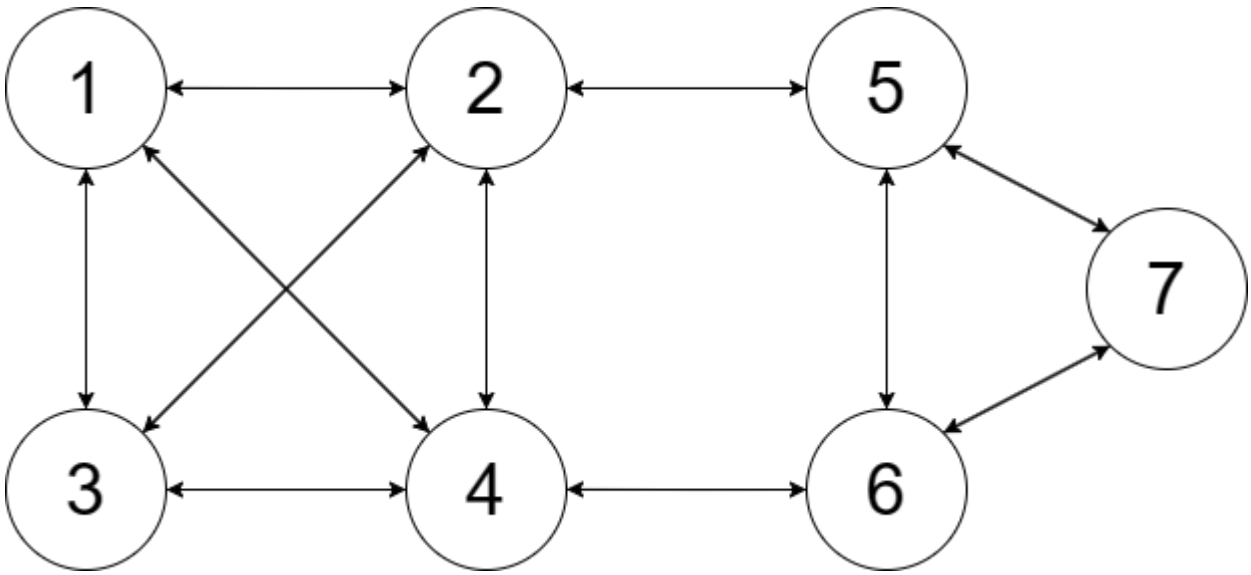
3. Implement the function `receive`. It should receive a packet and process it accordingly. It should log the events according to the sample output given.
    a. If the received packet is a message packet and it is not corrupted, it should save the packet to the buffer and send it to the next receiver. If this node is the final destination, it should send an acknowledgment/success packet backwards through the path.
    b. If the received packet is a message packet and it is corrupted, it should request a resending.
    c. If the received packet is a success packet, it should save the packet to the buffer and send it to the next receiver. If this node is the source, it should return an empty mac packet to end the simulation.
    d. If the received packet is an error packet, it should resend the previous packet in the buffer.

Network Class

1. Implement the constructor and functions `getNode`, `addNode`, `removeNode`. All connections are two-way, hence `addNode` should add the new node as a neighbor to the already existing nodes. `removeNode` should remove the reference from the other nodes.
2. Implement the function `calculatePath`. It should calculate the shortest path from source to destination and return a vector of nodes along the path.
3. Implement the `simulate` function.
    a. Calculate the path for the message to take.
    b. In an infinite loop, send the packet to the destination node. If the response is an empty packet, end the simulation. If the response is a message packet, corrupt the message with the given probability (Hint: `(float) std::rand() / RAND_MAX < corruptionRate`).

Main Function

1. Create the network given below.



2. Send the message “Hello world!” from Node 1 to Node 6. The expected output is below.

```
Message to send: "Hello world!". Path: -> 1 -> 4 -> 6
[Node 1] Packet received. Sending to Node 4...
[Node 4] Packet received. Sending to Node 6...
[Node 6] Packet corrupted: "Hello woYld!". Requesting resend from Node 4...
[Node 4] Resending previous packet to Node 6...
[Node 6] Packet corrupted: "Hello worldg". Requesting resend from Node 4...
[Node 4] Resending previous packet to Node 6...
[Node 6] Message successfully received: "Hello world!". Sending
acknowledgment to Node 4...
[Node 4] Acknowledgment received. Sending to Node 1...
[Node 1] Acknowledgment received.
```