CS 425 Assignment 2

General Instructions:

- You can write a program in C, C++, Java or python. (It must take input files name from command line argument)
- You must provide readme file which contains the language you used, the command for compile if required, run command and the version of language you used like Python 2.7 or 3.0.

Spanning Tree and Switches

For this program, you will be given two sample input files such as Input_1.txt and Input_2.txt.

An explanation for Input_1.txt:

First-line indicate the number of the graph (e.g. 2) followed by the number of links a graph is having (e.g. 5). This is followed by the links in each line showing how the switches are connected with each other such as **S2,1 S3,1** (indicating switch 2 port 1 is connected with switch 3 port 1).

An explanation for Input_2.txt:

First-line indicates the number of commands to run followed by the command itself. There are only three commands such as *Add_device, Send_Frame, and Remove_device*.
The command

- **Add_device 01 S2,3** implies a device with MAC 01 is added to switch S2 port 3
- **Send_frame 01 02** indicates a frame is sent from a device (host) with MAC 01 to the device (host) with MAC 02
- **Remove_device 01** means a device with MAC 01 is removed from the network.

1. Given a graph as input from Input_1.txt, implement the distributed spanning tree algorithm [Perlman 1985] to find the spanning tree as discussed in the class.
    i. You are required to print the periodic updates sent by each switches containing its *own mac address, root's mac address, hop distance to the root, and via the device mac address* in an output file (Named as Output_1.txt).
    ii. In each round of the algorithm, print the updates sent by the lowest mac address to the highest mac address.
    iii. You need to print the links of the final spanning tree in the Output_1.txt file. Again the order of the links should be prioritized by the ones that contain a node having the lower mac followed by the lower port number.
2. Using the network formed (spanning tree) in the previous steps and for each input from Input_2.txt

i. You are required to print the events happening in the switches such as (Backward learning by a switch) the update of port-address mapping, receiving a frame on In port and forward it to an Out port(s) in an output file (Named as Output_2.txt)

Note:

- the configuration of network devices are as follows:
  - (Switch: MAC address) = **S1:0F, S2:0A, S3:0B, S4:0C, S5:0D, S6:0E, S7:1A, S8:1B, S9:1C, S10:1D, S11:1E, S12:1F, S13:2A, S14:2B, S15:2C, S16: 2D, S17:2E, S18:2F** and so on.

Here is an example with an expected output needed to be printed in the output files.
Input_1.txt
1
5
S2,1 S3,1
S2,2 S4,1
S3,2 S4,2
S3,3 S5,2
S4,3 S5,1

Note: that links are bidirectional and appear in order (lower mac address followed by a lower port number for e.g. link S2,1 S3,1 comes before link S2,2 S4,1)

For the above graph, the periodic update sent by each switch must be printed in lexical order (lowest mac address e.g. S2 update is printed first then S3 update and so on) in Output_1.txt will look like

Round1
S2: 0A,0A,0,0A
S3: 0B,0B,0,0B
S4: 0C,0C,0,0C
S5: 0D,0D,0,0D

Round2
S2: 0A,0A,0,0A
S3: 0B,0A,1,0A
S4: 0C,0A,1,0A
S5: 0D,0B,1,0B

Round3
S2: 0A,0A,0,0A
S3: 0B,0A,1,0A
S4: 0C,0A,1,0A
S5: 0D,0A,2,0B

Once a spanning tree is formed, print the links of spanning-tree in Output_1.txt while maintaining the order in which the link appears in the Input_1.txt file, the links are bidirectional so there is no repetition.

For the above case, the spanning tree in Output_1.txt looks like
S2,1 S3,1
S2,2 S4,1
S3,3 S5,2

Now suppose, if Input_2.txt has the following set of commands to run

Input_2.txt
6
Add_device 01 S2,3
Add_device 02 S5,3
Send_frame 01 02
Add 03 S4,4
Send_frame 03 01
Remove_device 03

For the first command, Add_device 01 S2,3 the switch S2 will update its port address table to bind the mac 01 to port 3 once it received the first frame from this device, this event in switch S2 must be printed in Output_2.txt using the representation **S2 table_update 01,3**

In the case of Send_frame 01 02, the switch S2 will receive a frame on In port 3 and forward it to Outport 1,2 this event in switch S2 must be printed in Output_2.txt using the representation **S2 In 3 Out 1,2**

Note

- frame forwarded by S2 will be received by both S3 and S4 switch, follow the lexical order to print the events happening in the switch S3 first then S4 in the Output_2.txt.

In the case of Remove_device 03, switches will update the table via backward learning.

The final output for the above set of commands look like
S2 In 3 Out 1,2
S2 table_update 01,3
S3 In 1 Out 3
S3 table_update 01,1
S4 In 1 Out Null
S4 table_update 01,1
S5 In 2 Out 3
S5 table_update 01,2
S4 In 4 Out 1
S4 table_update 03,4
S2 In 2 Out 1
S2 table_update 03,2