## Question 3.

### 

The sketch is attached at the bottom of this report. B1 is a root port. Ports which have neither “d” nor “r” sign are disabled.

### 

B2 and B5 are placed at the same distance from B3, so we randomly assume that a message from B5 is received earlier than the one from B2.

The messages are received in the following order:

1. B3 sends: {B3, 0, B3} – B3 offers itself as a root
2. B3 receives: {B3, 1, B5} – B5 accepted B3 as a root because it has lower ID, and this is the BDPU message from B5. B3 sees that RootID is same as it’s own RootID, but B3 rootCost is lower than msg.rootCost, so it ignores this message. B3 declares itself designated on A.
3. B3 receives: {B2, 0, B2} – B2 didn’t accept B3 as a root. B2 offers itself as a root. B3 sees that msg.rootID is lower than B3 current rootID. B3 accepts B2 as root. rootCost is now 1 (portCost to B2). rootPort = port leading to B2.
4. B3 sends: {B2, 1, B3}
5. B3 receives: {B1, 1, B5} – B5 has acknowledged B1 as a root. Msg.RootID < rootID, so B3 changes: rootID = msg.RootID (1), rootCost = 1 + 1 = 2, rootPort = port leading to B5. rootCost > msg.rootCost, so B3 is not designated on A.
6. B3 receives: {B1, 1, B2} – B2 has also acknowledged B1 as a root port. But now:
   1. msg.rootID = rootID -> rootPort not changed
   2. rootCost = 2, msg.rootCost + portCost = 1+1 = 2 -> rootPort is not changed.
   3. rootCost = 2, msg.rootCost = 1, so port leading to C is not designated
7. B3 sends: {B1, 2, B3}
8. B3 receives: {B1, 1, B5} – nothing changed
9. B3 receives: {B1, 1, B2} – nothing changed
10. …

We end up with B3 having the following parameters:

RootID = 1, rootCost = 2, rootPort = <port to B5>, designatedOnLAN = {N/A}

NOTE: in the sketch in (a.), the rootPort from B3 leads to B2. This is equal, it depends on whether B3 receives packages from B2 earlier than B5. B2 and B5 have same rootCost, so the first to reach B3 will make B3’s rootPort connected to itself.

### 

Since the tables are empty, each bridge will broadcast the message to add it’s neighboring bridges/switches. So the message will be ‘heard’ on all the LANs in the network

### 

After the learning process, each bridge/switch knows on which port the computer on the A LAN network can be found. But because the computer on the I network never transmitted, no one knows where it’s located (on which port). So the message will be broadcasted on the whole network again.

### 

Now, each bridge knows at which port the computer at LAN A can be reached. Message from H reaches B4, received message on the same port, on which it has received message from A, so it drops it. Message reaches B1 -> LAN D -> B5 -> LAN A -> Computer A. B1 learned before on which port A is located.

So It goes through 3 LANS: H, D, A

### 

First and foremost, STP algorithm is here to avoid loops. If it’s not working correctly, loops will be created, which we will see here:

B4 stopped sending BPDU, so after a certain amount of time it will stop being the Designated Port for LAN I. And B6 will take its place. (B6 doesn’t know B4 exists). Which means, now LAN I has 2 bridges, where it is on a designated port.

Let’s imagine computer on LAN I sending packet to unknown computer on LAN B. Both B4 and B6 start broadcasting it. B1 also broadcasts it, and this broadcast is heard on BOTH B4 and B6. Which broadcast it again. And loop continues.

Even if stations on LAN I broadcast at 1% of the bandwidth, because of the loop, the packets stay permanently in the loop, which damages performance.