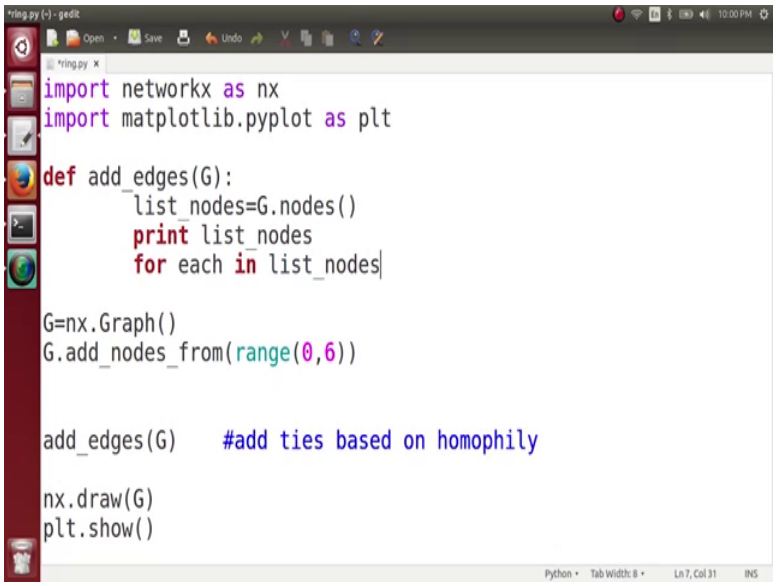
Social Networks
Prof. S. R. S. Iyengar
Department of Computer Science
Indian Institute of Technology, Ropar

How to go Viral on Web
Lecture - 152
Making homophily based edges

(Refer Slide Time: 00:07)

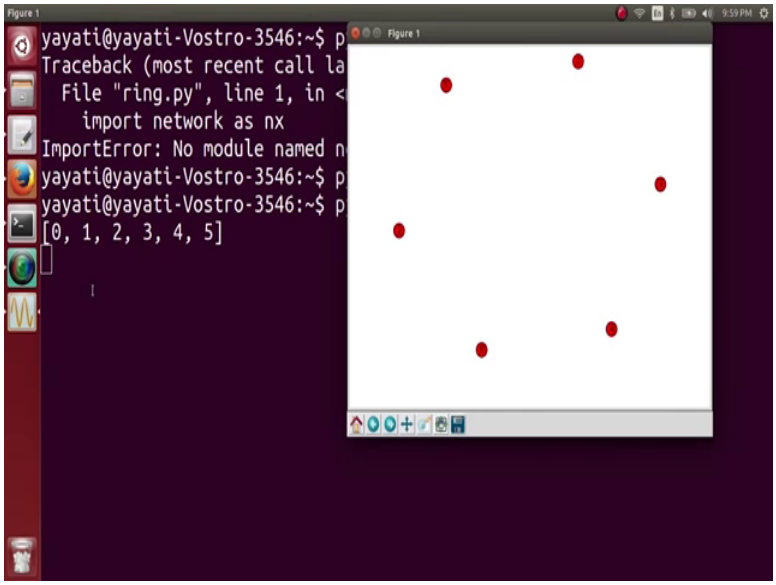


Next what we are going to do is, we want to add the edges corresponding to the homophily on this network and we have seen that according to the homophily, we are going to connect each node to 2 nodes towards its left and 2 nodes towards its right. And, we have also seen that when we were portraying this network, drawing this network the nodes were not in any particular order. So, for the time being to test the validity of the code I will simply change the number of nodes to 6.

So, we will be having 6 nodes number from 0 to 5 and next what I am going to do is I have to add homophily based edges to it. So, I call a function add edges G here. So, what this function is going to do is add ties based on homophily to this network and how do I do that. So, that is very simple. So, what I have to do? I have to connect node 0 to 2 nodes towards its right which are 1 and 2 and 2 nodes towards its left you can see that those 2 nodes are node minus 1 and minus 2, rather goes nodes are this node number 6 here and that node number 5. So, how do we do that?

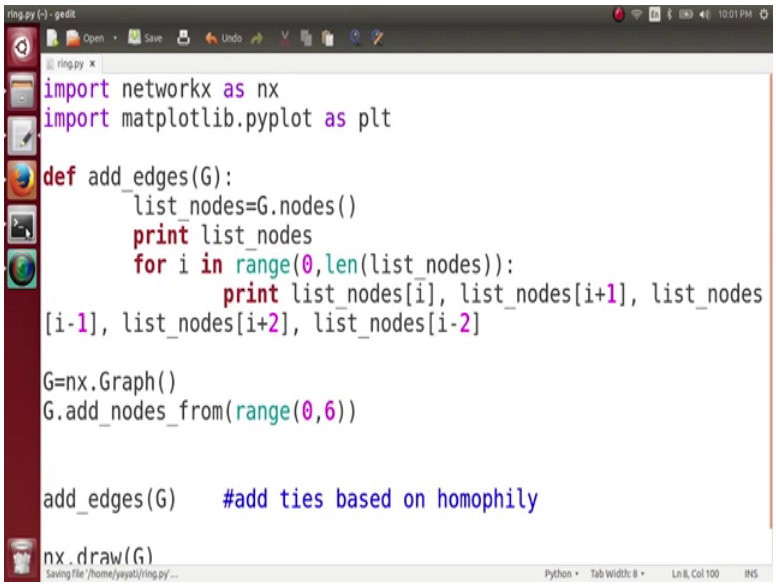
So, what we can simply do is we obtain a list of a. So, let me define the function here, define add edges and I will pass the graph G here and what I do is I obtain a list of nodes here; list of nodes equals to G dot node which are simply the list nodes equals to G dot nodes which are simply the nodes in this graph G. And, we can actually print these list nodes and see what it is giving to us.

(Refer Slide Time: 02:03)



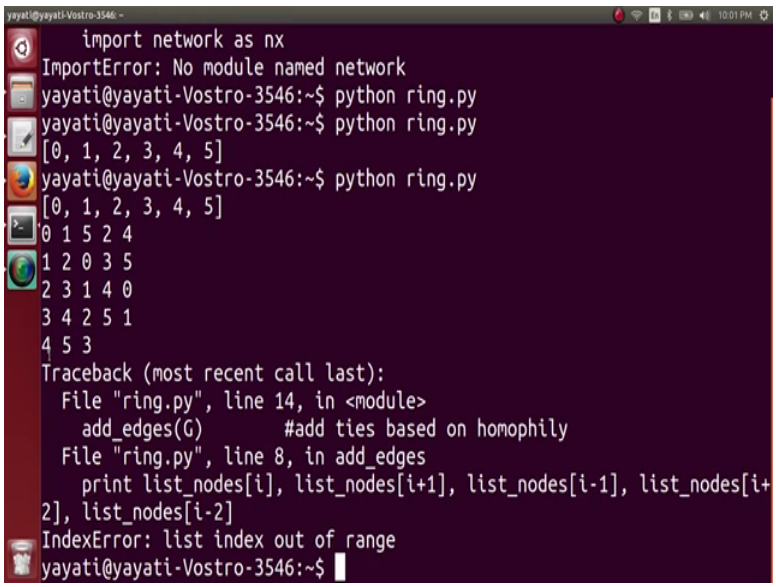
So, when I execute this code we can see that we got here 5 nodes 0, 1, 2, 3, 4, 5 well and fine. Now, what we have to do is ok, we have to connect each node. So, we have to take every element in this list nodes and every element is to be connected to 2 elements towards its left and 2 elements towards its right.

(Refer Slide Time: 02:39)



So, how do we do that is for each in list nodes, what do I want is I will print each and rather now each in list dot node for i a for i in range. So, we need to index in this is here for i in range 0 to length of list nodes ok. We are going to print, this we are going to print list nodes i right and where this list nodes i should be connected to is list nodes i plus 1, list nodes i minus 1, list nodes i plus 2 and list nodes i minus 2 right. So, this thing is very clear. So, we have seen previously whenever the in this is of a list goes in minus it start back from 0. So, let us see whether it is working or not.

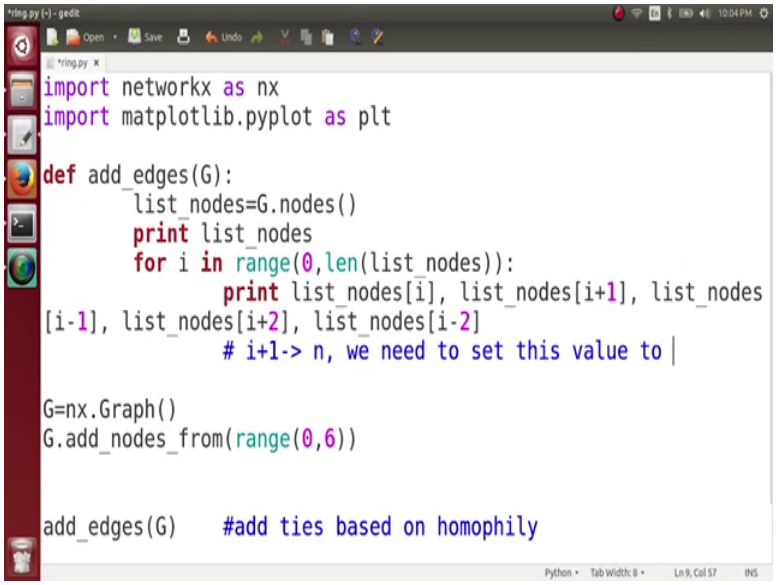
(Refer Slide Time: 03:47)



So, let us see what is it printing. So, you can see here 0 gets connected to 1 5 2 and 4, 1 gets connected to 2 0 3 and 5, 2 gets connected to 3 4 1 and 0, 3 gets connected to 4 5 2 and 1. Little bit problem with 4, it gets connected to 5 3 and then 4 gets connected to 5 and then 3 and then 5 3 and then 4 plus 2 is 6 ok. So, you so, you note the problem here. So, here when the index of a list whenever the index of a list goes in minus there is no problem, it starts up from the last element, but there is no element named as list nodes i plus 2.

So, what we can do here is little bit of manipulation is required here. So, list nodes i we can use and there can be a little bit problem with this i plus 1 and plus 2, i minus 1 and i minus 2 are perfectly fine. So, with i plus 1 and plus 2 there is a little bit of problem. So, we will just resolve that ok. How do we resolve that? When will i plus 1 first problem, you can say that i plus 1 will create problem when you go to this element.

(Refer Slide Time: 05:31)

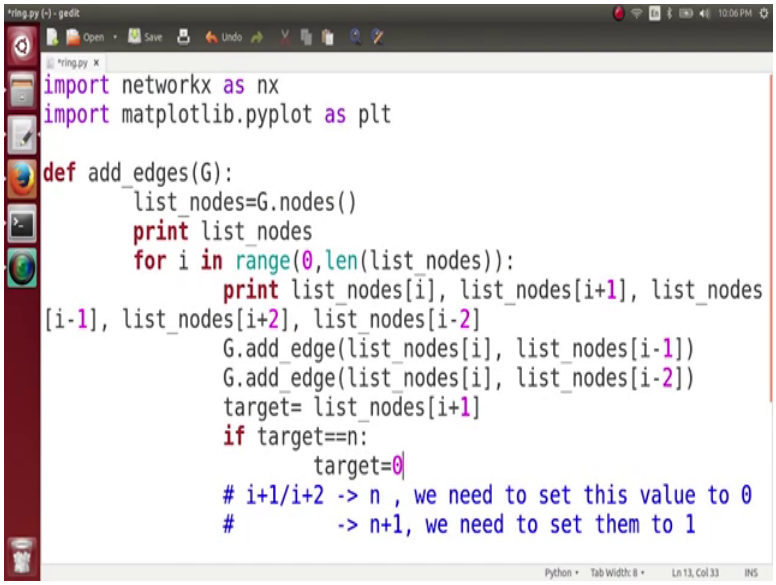


So, the last element of my list will be n minus 1, where n is the number of nodes. The last element is n minus 1 and it will create problem when it has to connect to list nodes of 5 plus 1 which is n. And, there is no list notes of i plus 1 because the length of the list is total length. So, we cannot access this element here. So, we can simply put a loop here.

So, what we are going to do is whenever are and then also now this one thing here before doing that list nodes of i is actually nothing, but i right list nodes of 0 is 0 list nodes of 1 is 1 and so on. So, whenever my list nodes of i plus 2. So, whenever i plus 1 has a

danger, whenever i plus 1 has a danger of becoming n, whenever i plus 1 has a danger of becoming n we need to set this value to what we go back from the starting.

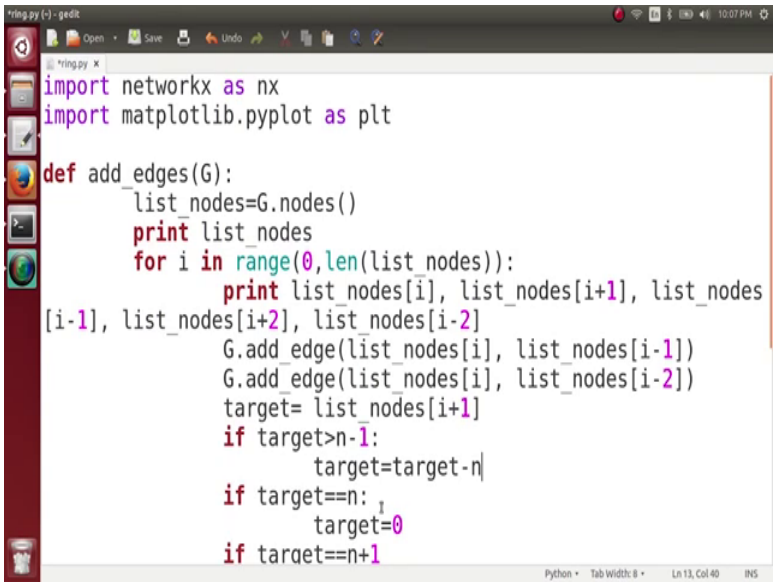
(Refer Slide Time: 06:57)



So, whenever it is n it has to be set back to 0 and whenever i plus 2 it tends to become ok, it can be n or whenever i plus 1 or it can also be i plus 2 whenever they tend to become n we tend to set that value to 0. And, whenever these two values they tend to become n plus 1 we need to set them to 1, who let us quickly see how we can set it. So, we can know that this node list nodes i has to be connected to list nodes i plus 1 i plus 2 i minus 1 and i minus 2, i minus 1 and i minus 2 are perfectly fine.

So, there can be a problem when we are connecting them to i plus 1 and i plus 2. So, what we are going to do is, what we have to actually do is we have to G dot add edge. And, from where to we have to add edge list nodes of i to list nodes of i minus 1 can be simply added and list nodes of i 2 nodes of i minus 2 can be simply added. And, now what we are going to do is let us say target list node of i plus 1 and what we are going to do is, if target equals to equals to n what we will do is target equals to 0 and rather ok.

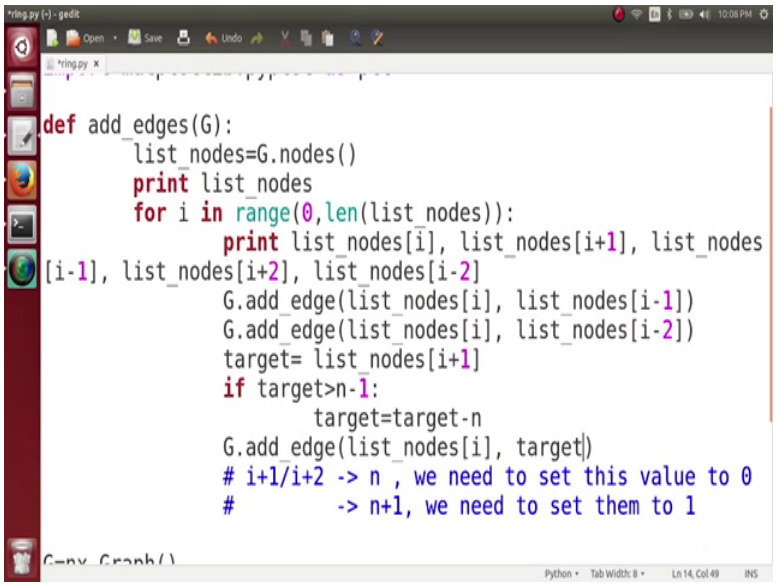
(Refer Slide Time: 09:13)



And then if target equals to equals to n plus 1, what we are going to do is target equals to 1; we can actually simply it. So, we can write both of these statements as 1 statement; what can that be? We can write if target is greater than n minus 1, what we are going to set the target to be? So, target is going to be set to, then I write target equals to n minus target. So, if target is greater than n minus 1, now if target is n it should be set to 0 and if target is n plus 1 it should be set to 1.

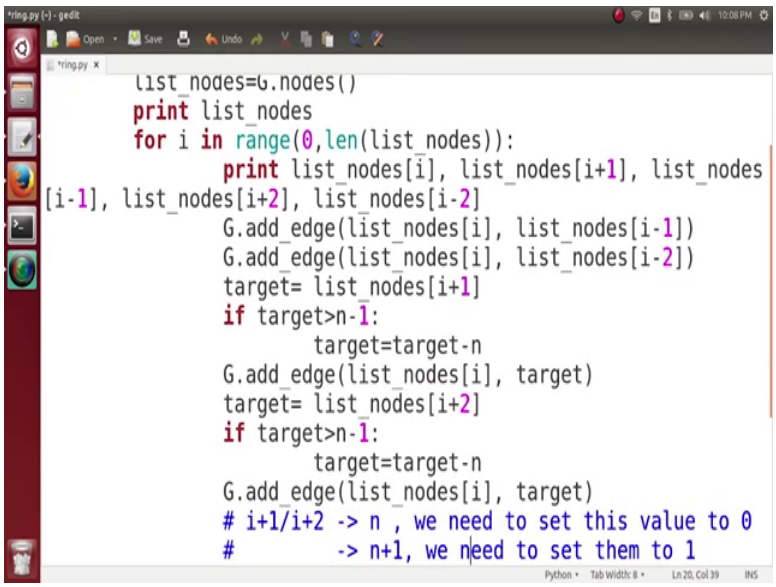
So, target is nothing, but target minus n. So, can I do target equals to target minus n, what will we do if target is n it will set target is 0 and if target is n plus 1 target will be set to 1 ok. So, target equals to list nodes i plus 1 and then this happens.

(Refer Slide Time: 10:27)



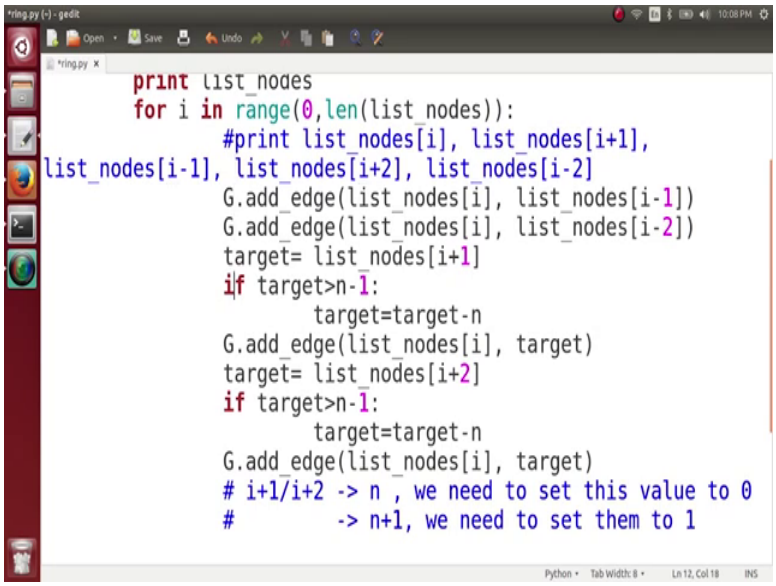
And then what I can simply do is G dot add edge from list nodes i to target right and now this target can be list nodes i plus 1 and this target is also going to be list nodes i plus 2.

(Refer Slide Time: 10:53)



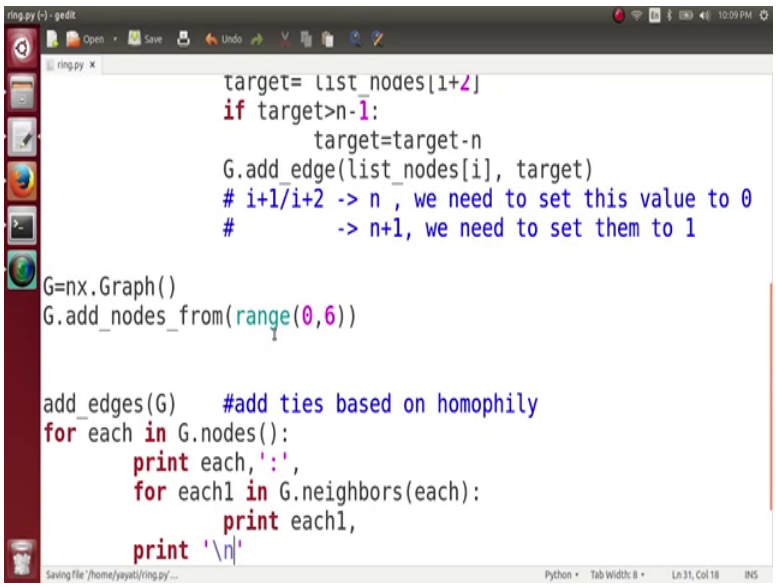
So, essentially going to replace this whole thing with if target equals to list node of i 2 and then this code remains the same and, then G dot add edge list nodes i comma target right and then we can comment this statement.

(Refer Slide Time: 11:09)



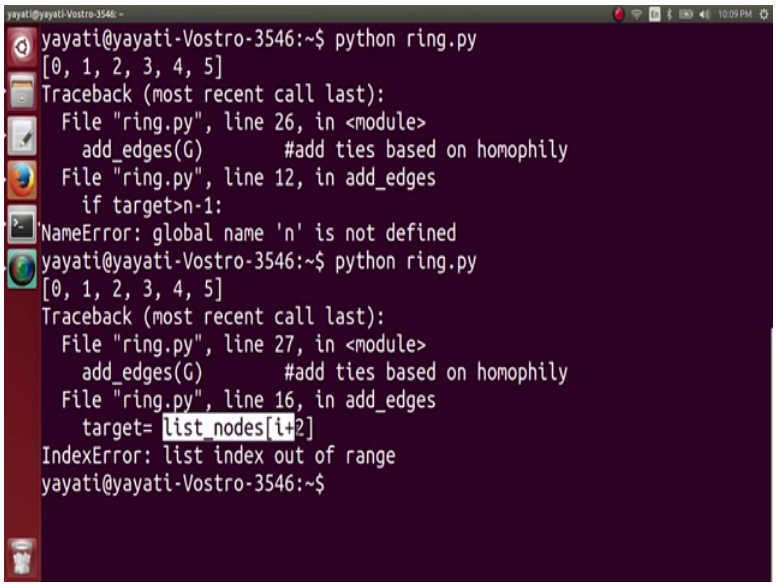
Now, to see whether this code is working fine or not, what I am going to do here is I am going to look at the neighbors of each nodes.

(Refer Slide Time: 11:13)



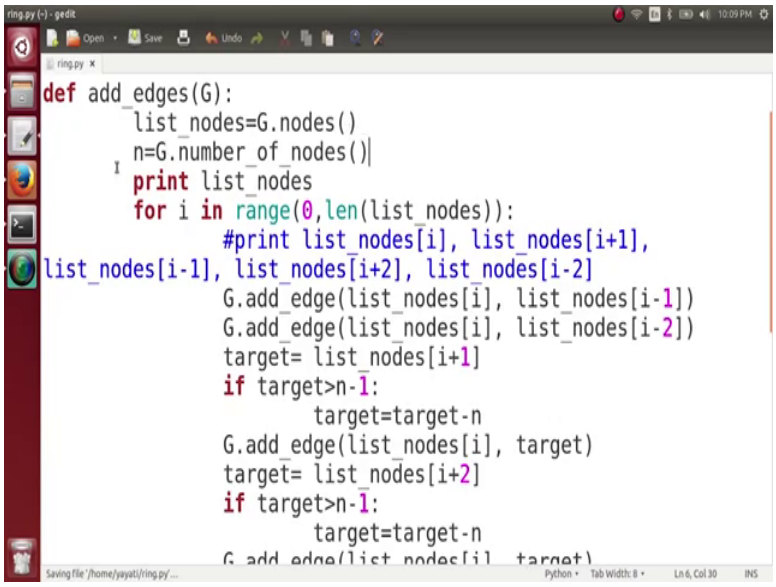
So, for each in G dot nodes I am going to print each comma comma and then for each 1 in G dot neighbors I am going to look at all the neighbors of this node each here. And, I am going to print this neighbors one by one, print each1 then I print the new line here; let us now execute this code and see ok.

(Refer Slide Time: 11:53)



So, we get an error name n is not define. So, we have again and again used n here and we have not define what is n. So, it is easy what is n as we have discuss, n is nothing but the number of nodes in the network.

(Refer Slide Time: 12:09)

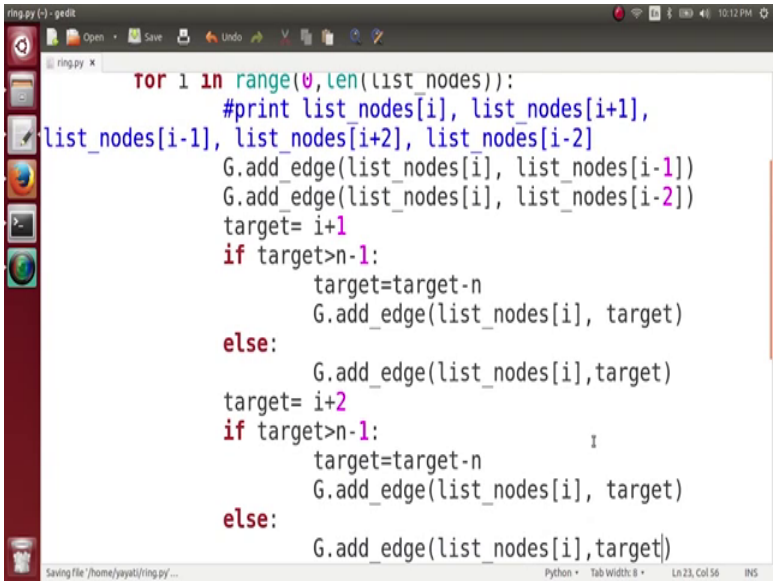


So, n is G dot number of nodes ok, here is again some problem target equals to list nodes i plus 2 line number 16 ok. So, what is going to happen here is when we obtain the target equals to list node i plus 1 just simply showing as an error here. Now, what can we do

here is will be here is remover. So, we will node add list nodes of i is namely nothing, but i until and unless this value here becomes in negative.

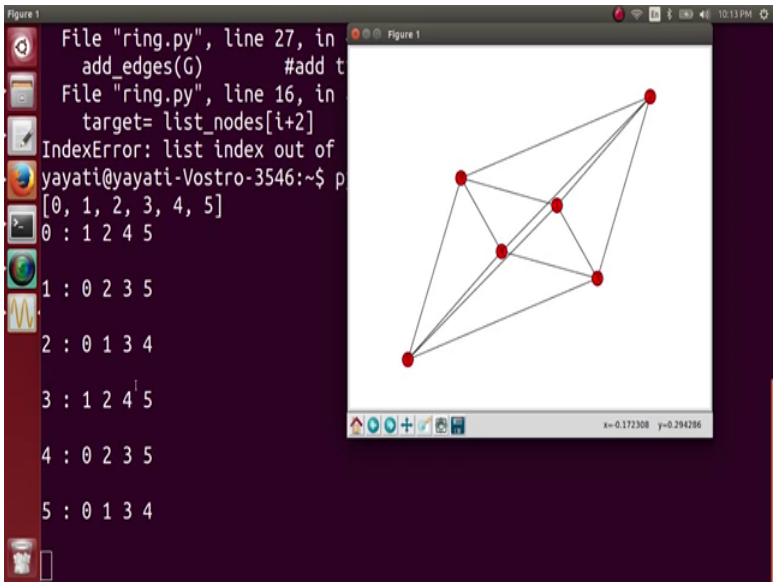
So, list nodes of i minus 1 did not be i minus 1, it can go back and start from the last element in list, but list nodes of i is essentially going to be i. Now, what I can do here is instead of list nodes of i plus 1. So, the maximum index in this list is n sorry n minus 1. So, if it the here becomes the value n so, it cannot determine the value for list underscore nodes n. So, what I am going to do here is target equals to i plus 1.

(Refer Slide Time: 13:39)



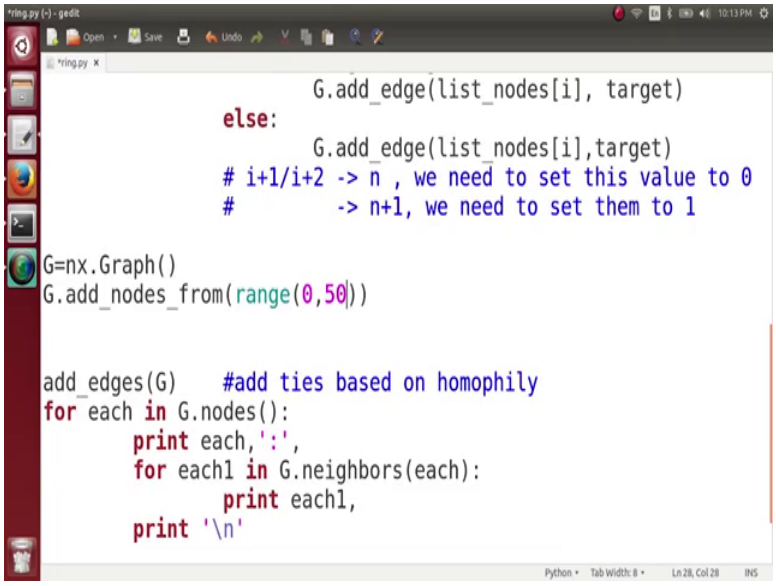
So, if target equals to i plus 1. So, now, if target is greater than n minus 1 we are going to change the target to target minus n and then we are going to add the edge. Otherwise what we are going to do is, else G dot add edge otherwise there was no problem and we can simply add an edge from list nodes of i to target. And, similarly here so, what I am going to do here is target equals to i plus 2 and if target is greater than n minus 1, this things happens, else we can simply put an edge from list nodes i to target without changing target. So, that has now run this code and see ok.

(Refer Slide Time: 14:51)



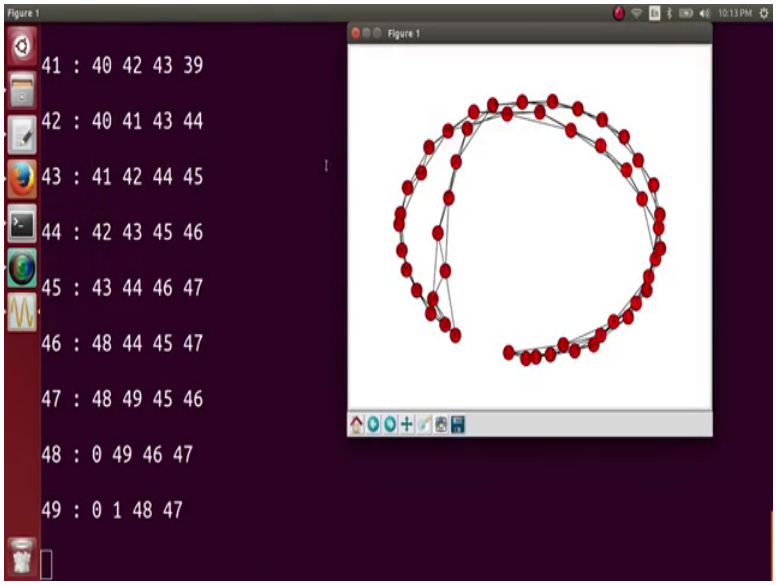
So, we say here we got a graph here 0 is connected to 1 2 and then back towards 4 and 5, 1 is connected to 2 3 0 and 5, 2 is connected to 0 1 3 4, 3 is connected to 1 4 3 is connected to 1 2 4 5, 4 is connected to ok; little bit problem with 4 here. 4 should be connected to 3 2 0 and 5 right, yeah 4 is connected to 3 2 0 and 5 perfectly fine. And, 5 should be connected to 2 towards its left which are 3 and 4, 2 towards 6 right which are 0 and 1. So, this is perfectly fine. So, we have created the homophily based links.

(Refer Slide Time: 15:43)



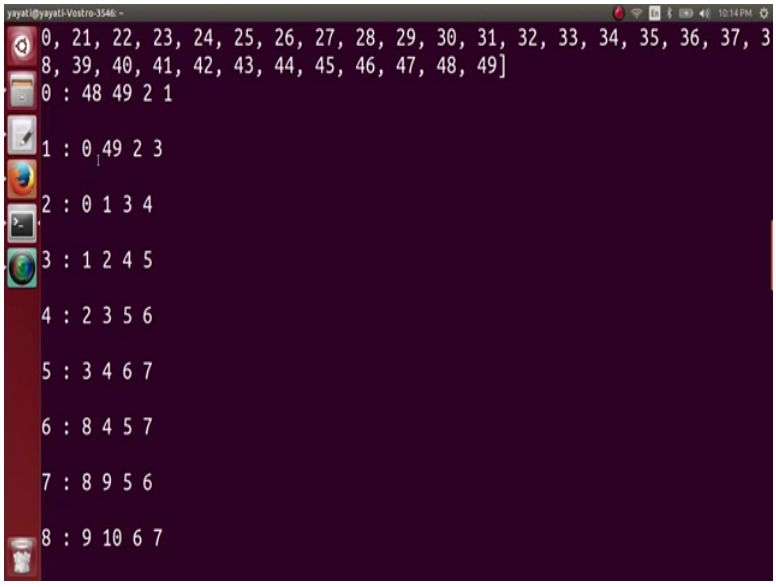
And, what we can do is we wanted a graph here for 0 to, let us say we want 50 nodes here. So, we I will make it 50 here and then we can execute it and see.

(Refer Slide Time: 15:47)



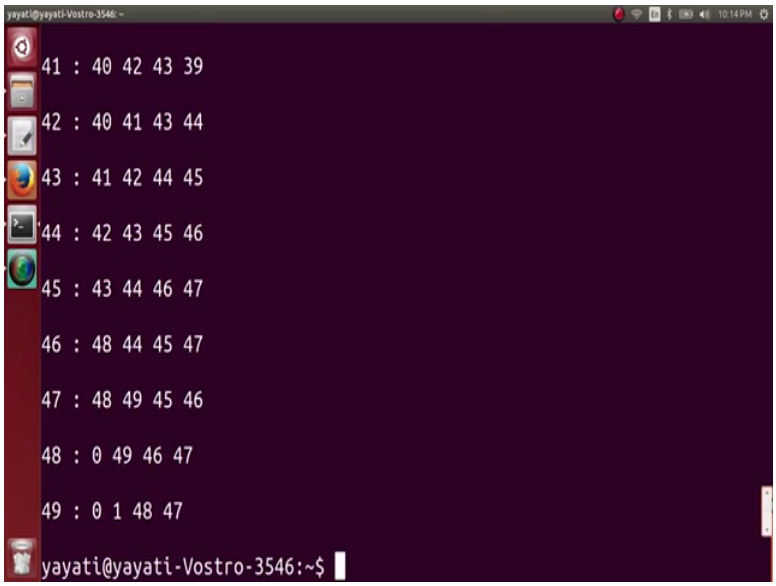
It might look here that this graph is ok, you cannot see a complete cycle here, but you can test the edges here and ensure that they all are correct.

(Refer Slide Time: 15:57)



So, you can see that 0, 1 is 0 is connected to 1 to 48 and 49, 1 is connected to 2 3 0 and 49, 2 is connected to 0 1 3 4.

(Refer Slide Time: 16:11)



So, you can verify all these edges. So, verifying all these edges you conform that we are getting a perfect cycle here, where every node is connected to 2 nodes towards its left and 2 nodes towards its right.