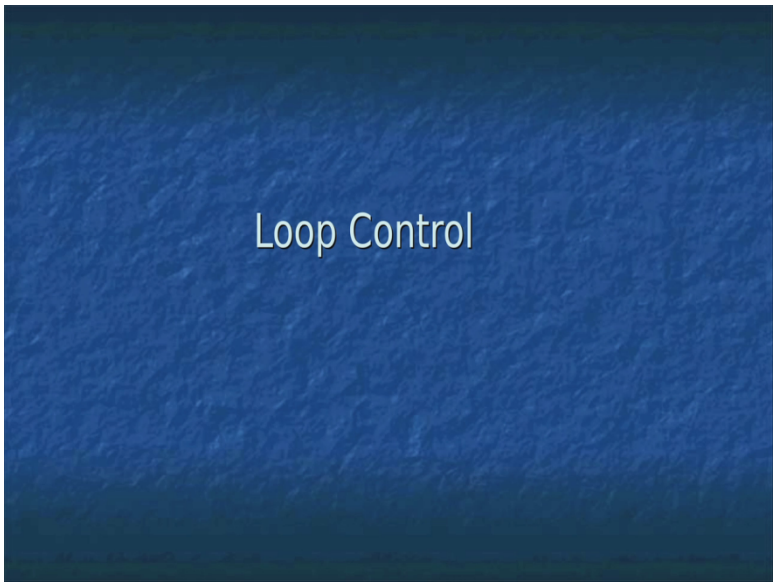
Information Security 3
Sri M J Shankar Raman,
Consultant Department of Computer Science and Engineering,
Indian Institute of Technology Madras
Module 35
Shell Loop Control

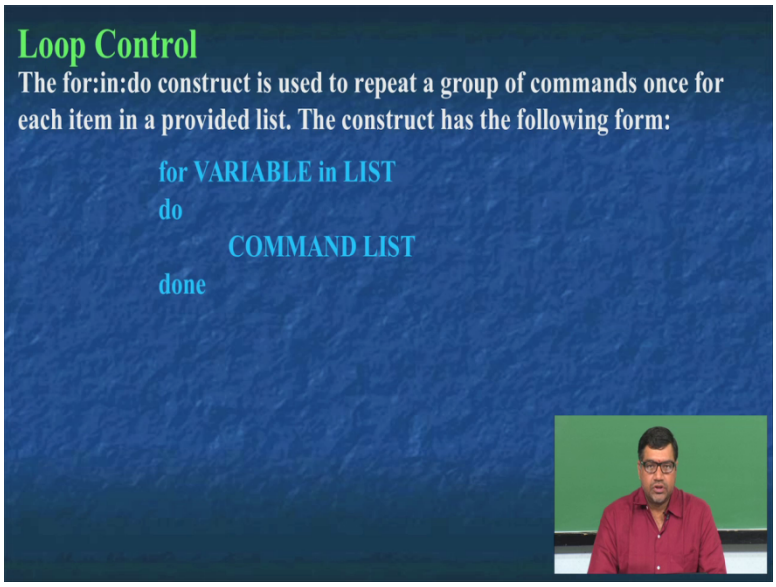
In the last section we saw how to use if then else statement. We also saw how to do the file test operators. In this session we also found out that we had a problem in the previous program where we gave two files and tried to find out what are properties of those two files unfortunately our program could not identify or know about the second file and therefore what we did was it only took the first parameter and it actually dropped the second argument what was given and of course the program worked correctly for the first argument that you had given.

(Refer Slide Time: 01:02)



What we will do in this section is, we will try to go into the concept of loop control thereby we will be able to find out the characteristics of all the files that are given as arguments.

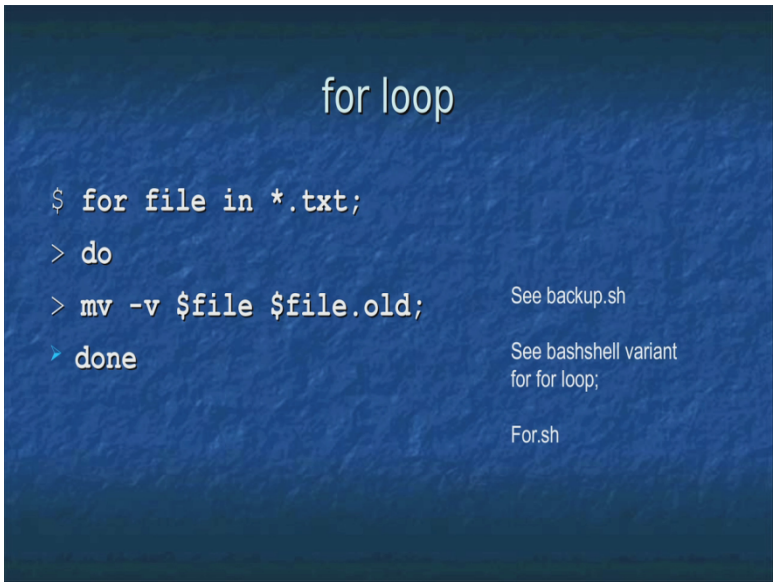
(Refer Slide Time: 01:25)



Now this first loop control mechanism we are going to look at is the for loop ok? And the concept we are going to look at is for:in:do construct which is used essentially to repeat a group of commands once for each item in the provided list. say for example in our previous case it could be the argument list that we are giving.

So the syntax for this loop control is like this we have to type for followed by whatever variable you want to use and then you have to follow it with the word E and then you can give the list of files or the list of items that you want to work upon. And after that it should be followed by the keyword do and you can give a bunch of command list I mean whatever command followed by the list or the variable name. And then finally you have to end it with done.

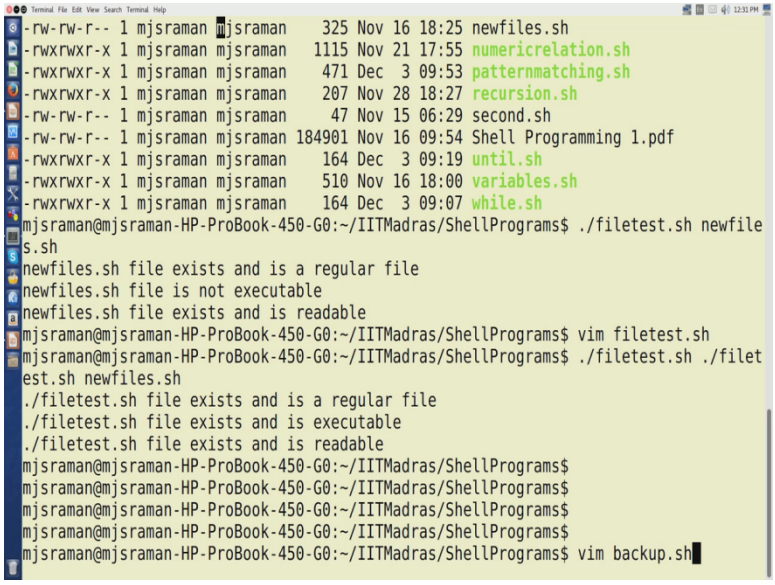
(Refer Slide Time: 02:31)



So now what we will do is we will try to take an example and then see how this construct is generated so here is an example what we have so we have for we know that is the keyword and then file is the variable and then in and if your remember star dot text means any file that ends with txt as the last three characters and then what we do is we move the file with a minus v option; minus v option tells you that we have to move in a mode which tells you that whenever you execute a command you just say what you are doing. And finally it closes with a done.

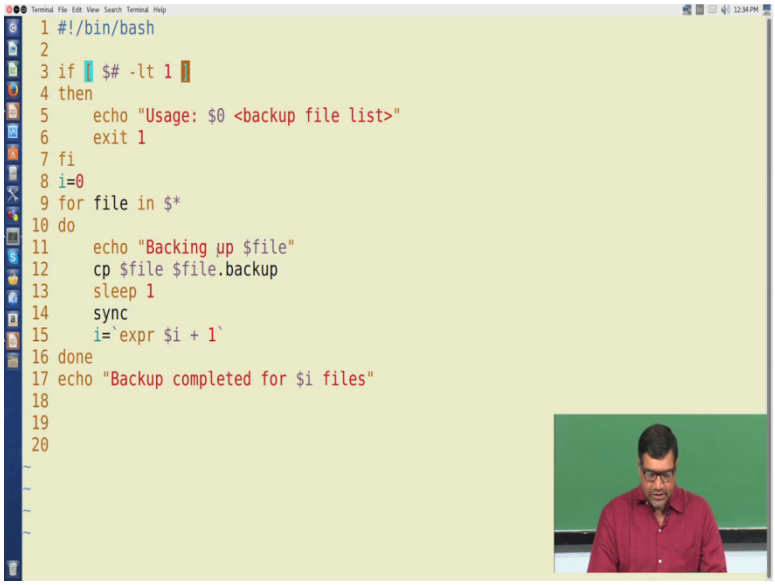
Now what this statement will actually do is? it will take all the files that end in star dot txt and then convert them as star dot txt dot old. Now let us see this with an example. Let us take a look at the file called backup dot sh.

(Refer Slide Time: 03:41)



So we have this file called backup dot sh, Ok?

(Refer Slide Time: 03:47)



What we do is? In this file if you look at this we have this line number 9 we have for file in dollar and then star. So I hope you remember from your previous classes that dollar star brings all the arguments that are passed to this program. So in the form of this list it brings all the arguments here after the in and essentially what happens is that so this value of file ok this file variable takes one element of that list and then if you see this command called cp we know that it is the copy command it converts this file from the current name to current name

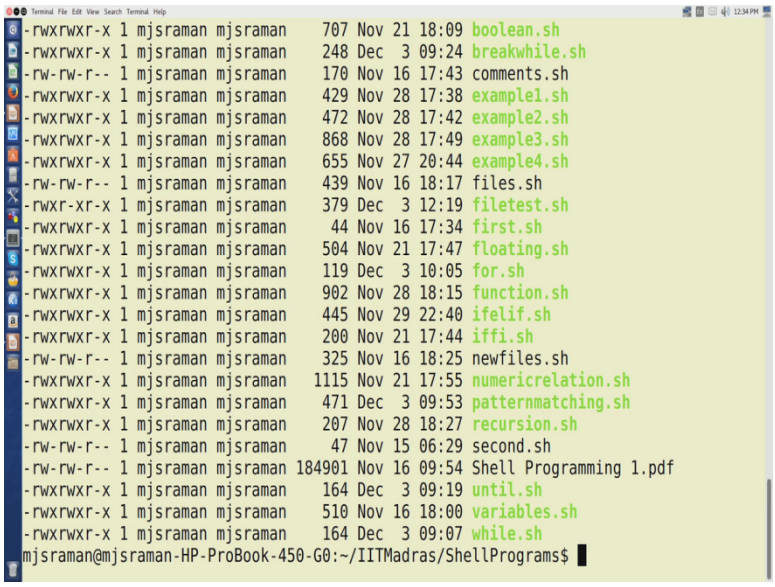
dot backup and let us now this is the core syntax ok? And then finally we are closing this with done.

And let us try to read this program completely so what happens in line number 1 is that we are invoking the Bash cell and then in line number 3 as usual we are actually checking whether this the number of parameters that you pass for this is correct ok?

So in this case I should have at least one argument that is passed to this function and then what I do is if I do not pass that one argument it just says that usage is how to use this file and then finally what I do is I initialize the variable of I to a value of 0 and essentially what I try to do is? I try to keep track of how many files I am moving into backup, ok? And of course this sleep 1 tells you that after moving this you have to sleep for 1 second.

Essentially this sleep command tells you that you have to wait I mean the reason why you are doing it otherwise the program moves very fast and it is very difficult to observe what is happening. Therefore what I do is I put this sleep command and then the syn command tells you that you have to write it into the buffer whatever you do here you write it in the buffer I mean you have to write it back to the disk ok and then this thing we know that we are trying to increase the value of i by 1. So essentially what we will do is we will now try to run this command.

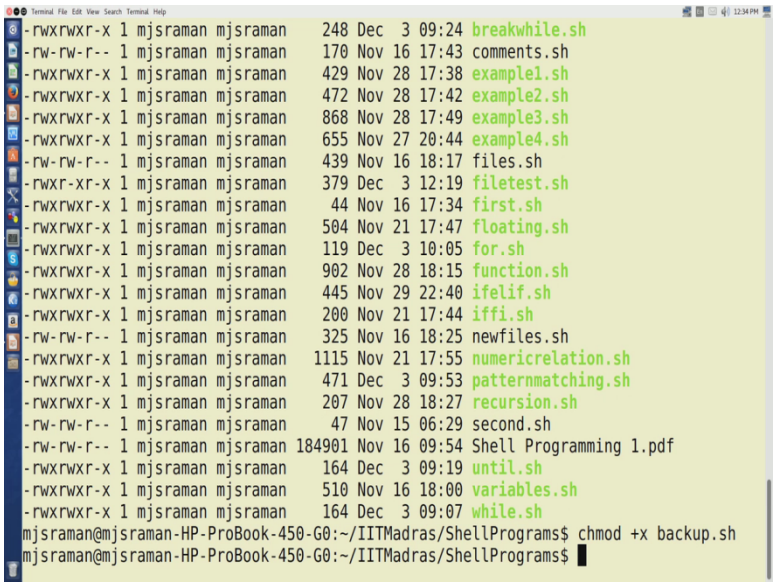
(Refer Slide Time: 06:22)



And what we will do is if you look at this directory we have lot of files ok the slash dot sh and let us assume that I want to take back up of all these files just for ensuring that we do not

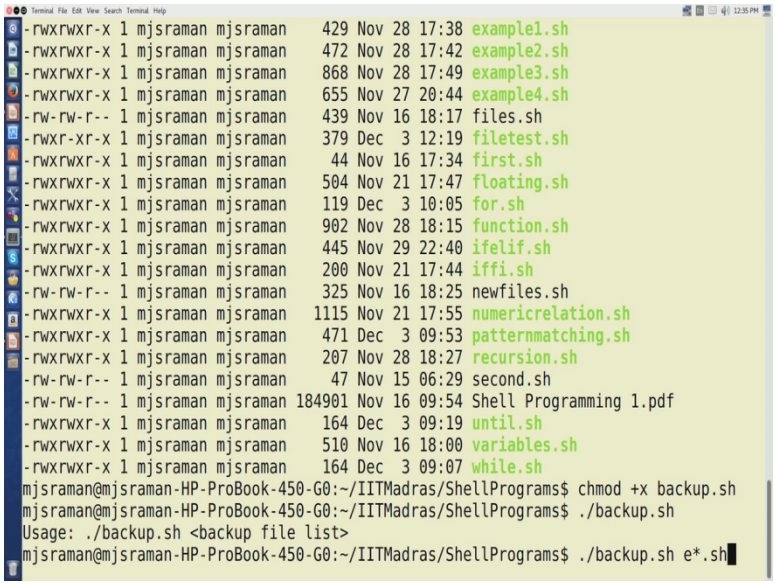
loose these files. So what I can do is instead of saying that copy a function dot sh DOT bak what I will do is I will try to run this program. So if you look at this program so what we will do is we will run this backup dot sh.

(Refer Slide Time: 06:55)



And let me change the mode of this.

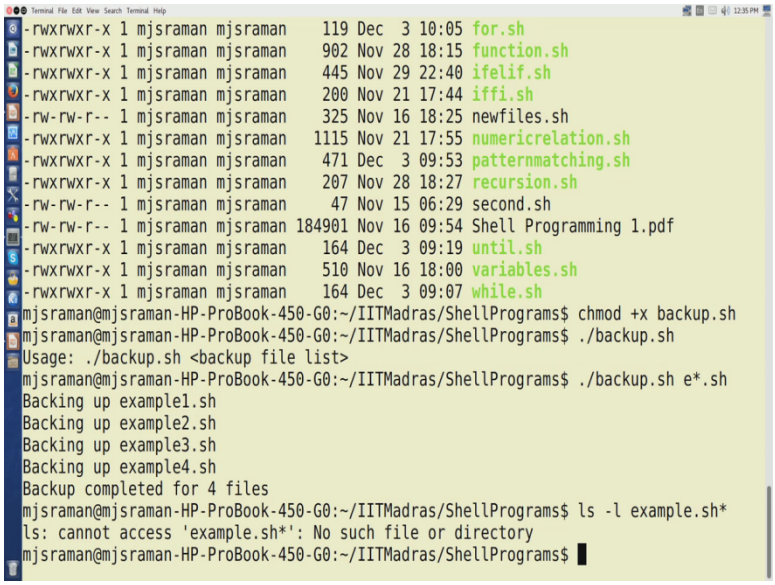
(Refer Slide Time: 07:01)



Then let me run this program, so let me try to find out the usage we are just typing this command so it just says that you have to give the backup file list. So as I told you we can do

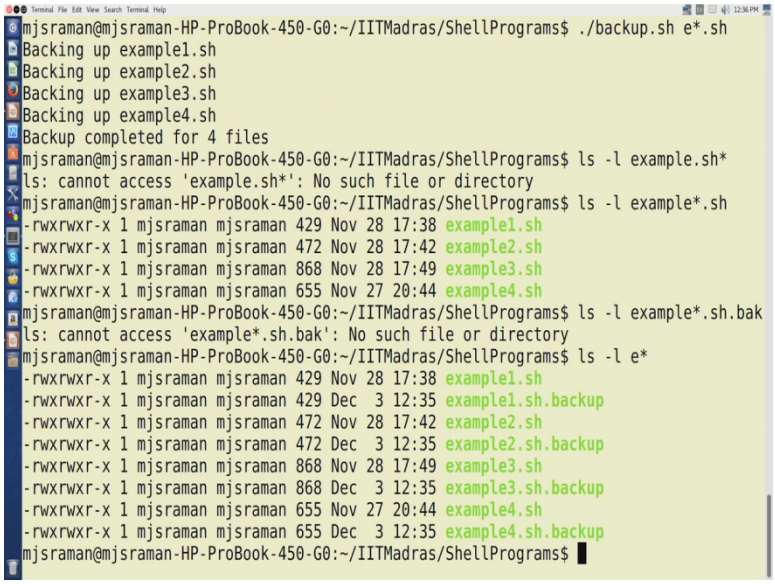
backup dot sh then I can say start so let us assume for the time being we want it take all files that start with the e and then I want to back them up.

(Refer Slide Time: 07:22)



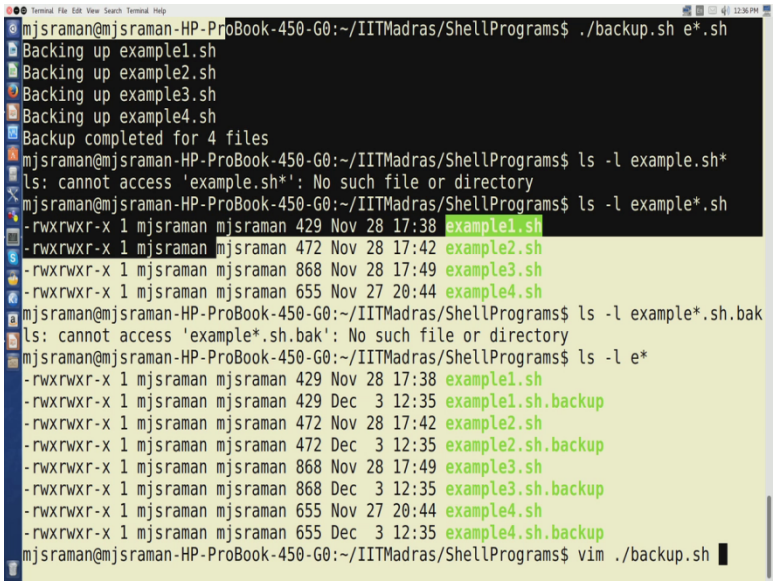
So I am saying e star dot sh. So all the files that start with the e and that end with sh I want to backup. So if you look at this so this program says that it is backing up example1 dot sh it is backing up example2 dot sh and it has backed up 4 files ok?

(Refer Slide Time: 07:44)



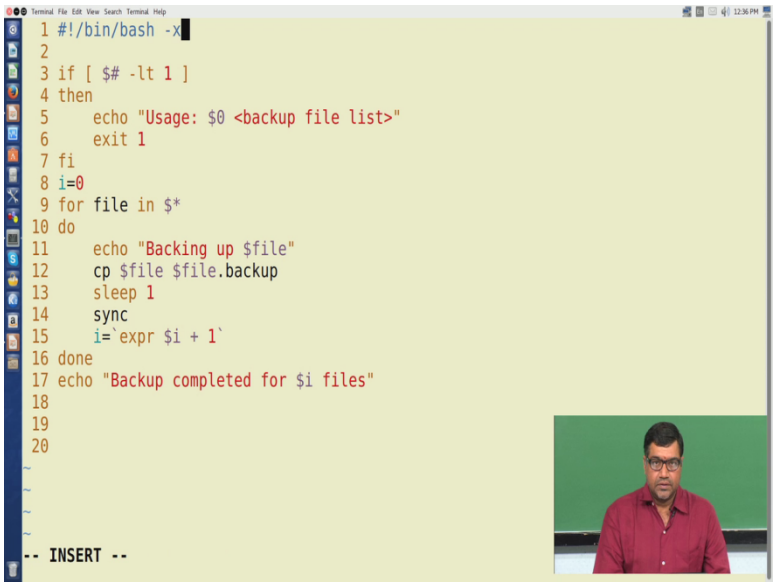
So ls minus l example star dot sh ok we have these 4 files and let us say we have dot back let us see e star

(Refer Slide Time: 08:03)



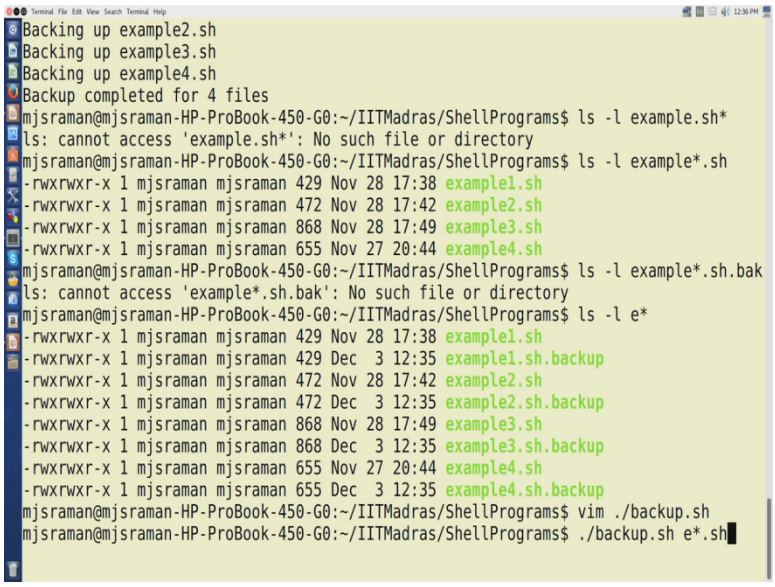
So if you look at this each of the file you have a backup so in this case we have let us see how this program works so we know that the sh minus x will let you show you how the program will work so what we will do is we will take the backup dot sh file

(Refer Slide Time: 08:22)



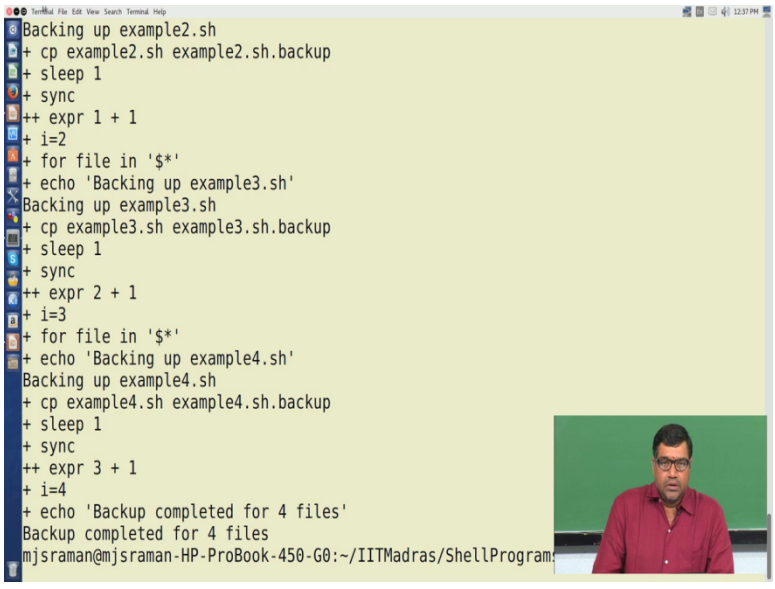
And then we will put the minus x option and then let us see how this program works.

(Refer Slide Time: 08:28)



So we will add the minus x option goint to tell you how the program can work? Ok. So what we will do is we will just run the backup dot sh the same way as we run it last time so backup dot sh e dot sh

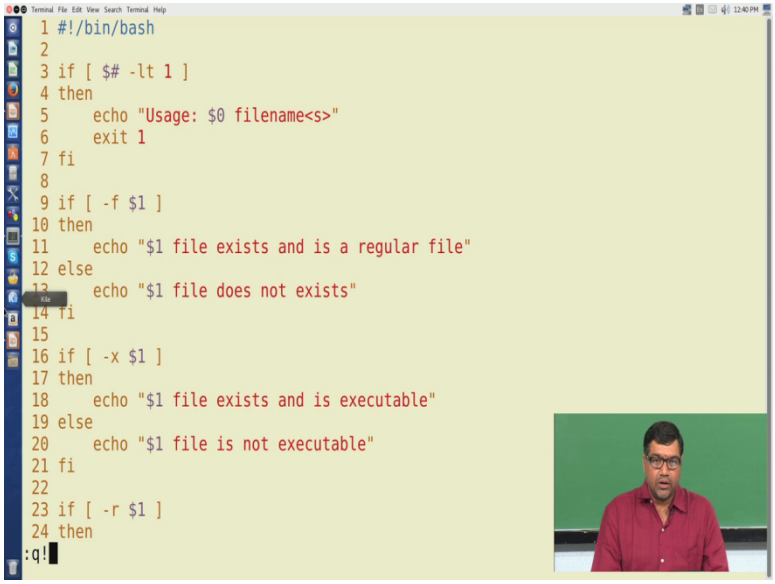
(Refer Slide Time: 08:42)



So if you look at this so let the program run and then we will get the explanation on how this works. So let us take this example ok so we are coming to backing up example3 dot sh. So in this case the statement i is equal to 2 ok and then what we do is for file in dollar star so it tries to so the example will restore so in this dollar star will restore all the files that is example 1

dot sh example 2 dot sh, example 3 dot sh and example 4 dot sh. So in this third loop since we are counted i is equal to 2 so i is equal to starts with 0 , i is equal to 1 and i is equal to 2 so this is third time it is executing therefore it is taking up this file called example3 dot sh and it tries to back it up so if you look at the for loop it just says for f ile in dollar star. So it is dollar star now use example 3 dot sh then after that it backs up and then it is sleeping for a second and then it is syncing up ok and then finally it is incrementing value of i and then finally again goes to file number 4 which example 4 dot sh and once example 4 dot sh which is the last element in the list once that comes through then this program ends and prints the message that backup completed for 4 files.

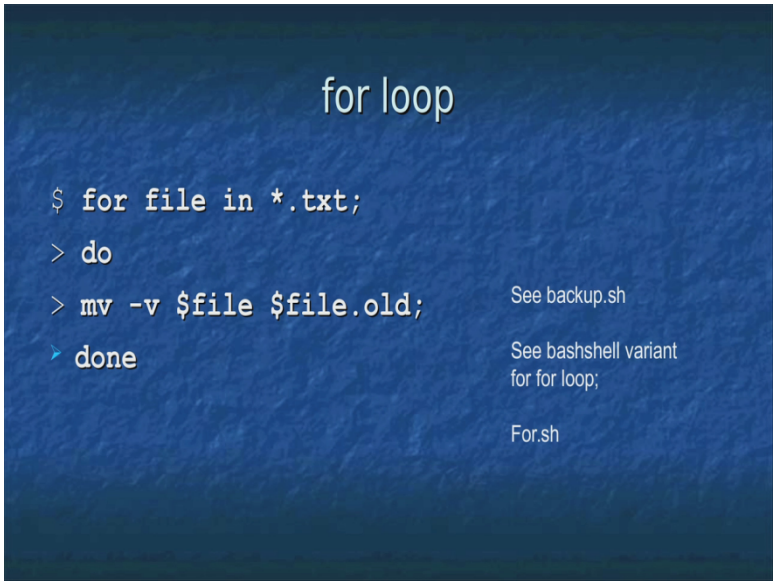
(Refer Slide Time: 10:00)



So in file test dot sh so what happens if I replace this by dollar star so that was exactly what the for loop was doing ok. Now what would happen is if you replace it by dollar star this will have a full list of file so it could lead to a syntax error so let us try to do this ok I put a dollar star and then trying to execute this program file test dot sh now if you look at this ok so I have to give a file name so let us say e star dot sh. Now if you look at this there are too many arguments so this error occurs because if you look at this file test dot sh this dollar star gets a list because I put dollar star then it gives all the files example 1, example 2, example 3, and example 4 and if you know in this condition you should give only a file name therefore it has list of file therefore it gives an error, so what we should do is? this whole code we could change it by putting a for loop and if you want to give a bunch of files say for example usage is something like this filenames, instead of filename let us say I want to give a file names ok.

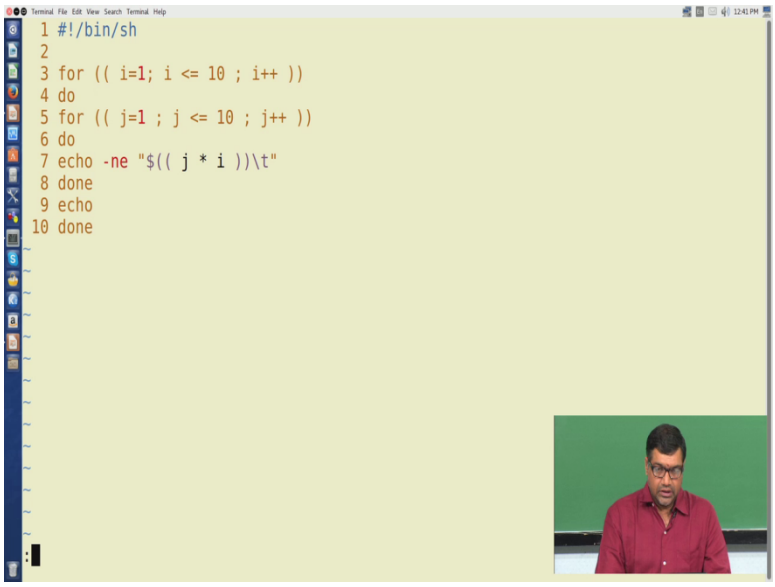
Then what you could do is you could use the for loop that we had shown in the previous example and using this for loop you could have printed the values of each one of the files.

(Refer Slide Time: 11:34)



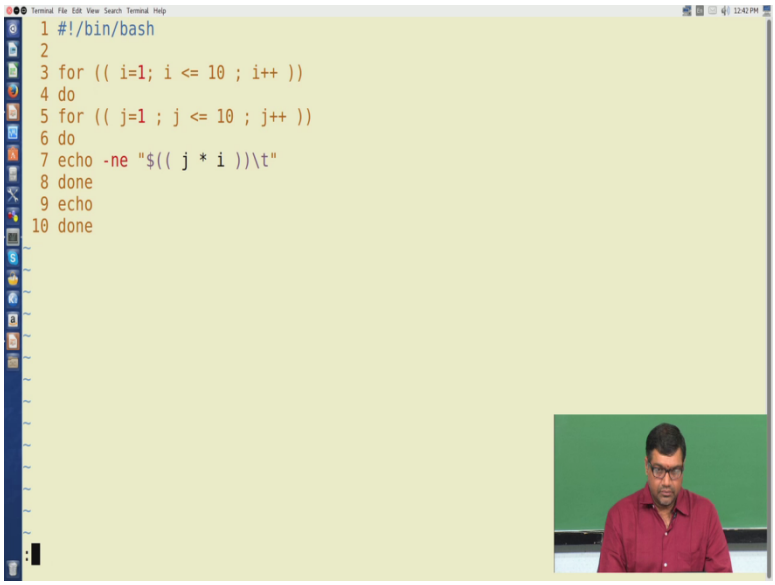
So this is one of the use of the for loop. Now we will take a very slight deviation and look at a bash shell variant of a for loop. So essentially what happens is that if I mean this is just to satisfy the curiosity of people who already knows programming using C programming language.

(Refer Slide Time: 11:55)



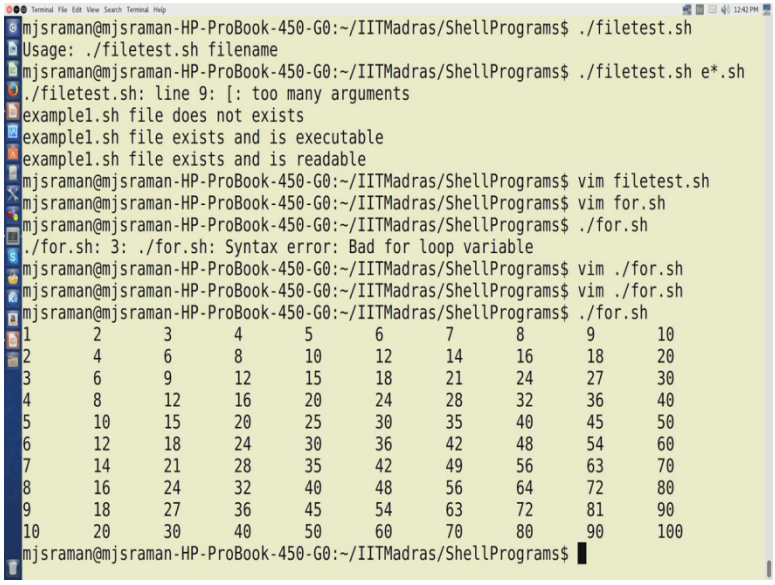
So what we will try to do is we will try to see one more variant of bash shell ok where this for loop see until now you are being using the for loop for doing the file operation so if you look at you are to take a list of files and then the find found a list. Now is there any variation in Bash shell for having a c like structure for repetitions so the essentially we are talking about loop control and if people who are familiar with C programming language will know that in Bash shell we have a command similar to the C programming for loop so here is this command ok where we try to have the for loop like this so if you look at the syntax of this it says two curl braces and then inside that I give i is equal to 1, i less than or equal to 10, i plus plus which is in the same form of C and if you look at this program this works only with the Bash shell because if I give sh.

(Refer Slide Time: 13:00)



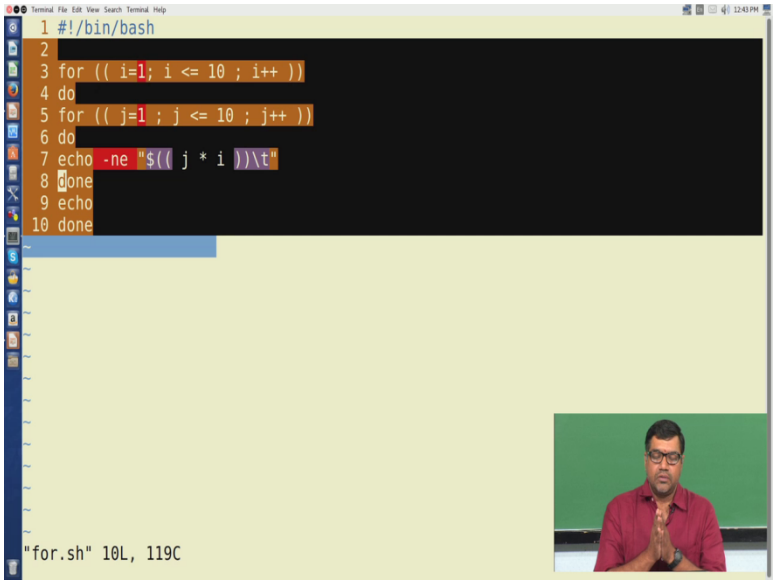
Let us try to execute this program with sh what happens is that it gives a syntax error ok ? So whereas if I go and execute this program with a Bash shell so if I execute this program with a Bash shell then let us see what happens.

(Refer Slide Time: 13:20)



So now it prints the shell. So this is one or two things that I wanted to tell you that this course you we will look at both shell and Bash shell and these are some kind of minor variations that are found in Bash shell ok? So most of the programs that we run in the course will work with shell and Bash both. But then there are examples like this that you will have to use Bash shell specifically. So let us take a look at the syntax of this loop control statement. ok?

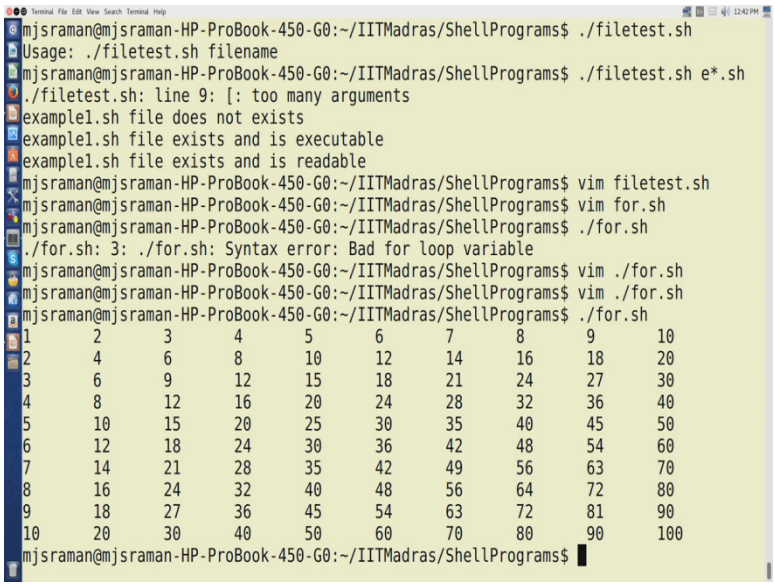
(Refer Slide Time: 13:55)



So this is similar to C programming loop so you have to give the initial condition the initial condition is i is equal to 1 and then the termination condition is (i is less than or equal to 10,

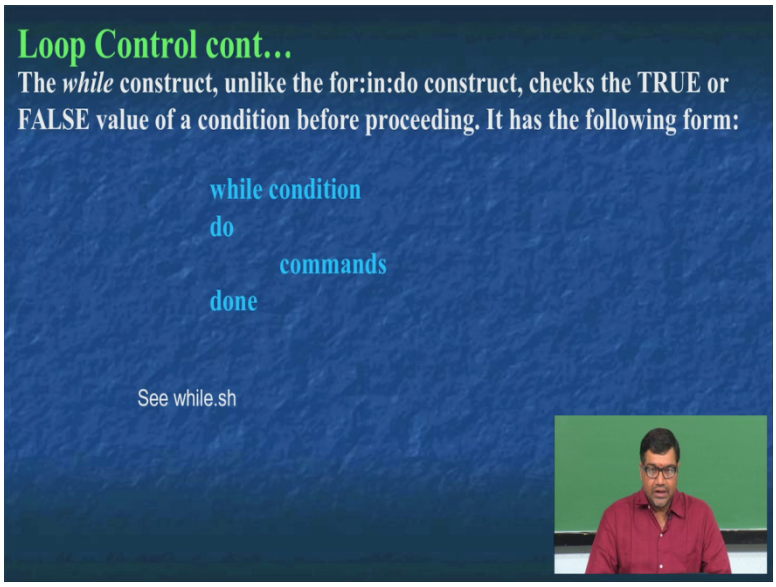
And then the increment value is i plus plus). So similarly so you can also see that we are having a loop within a loop and this is also allowed so just like if statement and within the if statement you can have an if statement or within an else statement you can have an if statement that we saw earlier, within a for loop here. And if you look at this so in this case I am using a tab to print the tables ok and this is one more variation of course we will not use this variation in many of our exercises but I just wanted to show you that a loop control is not only applicable essentially for text files it is also applicable for this kind of numeric calculations.

(Refer Slide Time: 14:48)



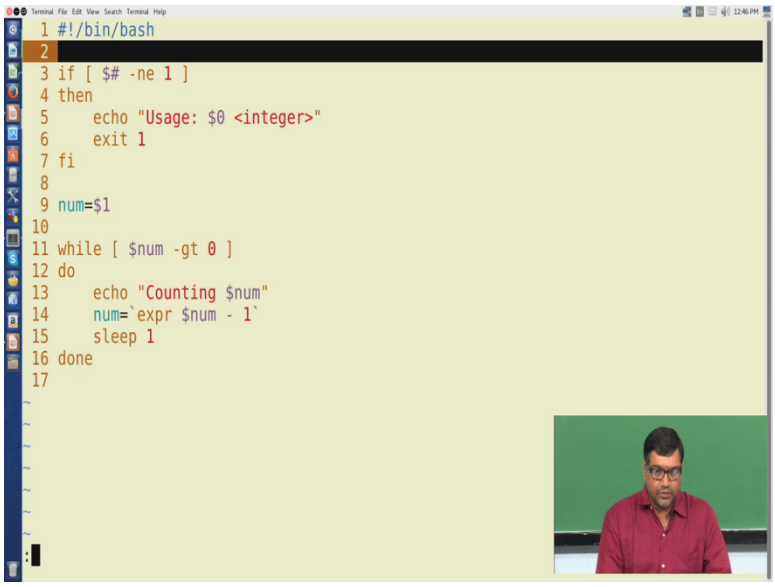
So other than this is not of much important to this but the idea is to show you that there are two variants of for loop in the Bash shell the first variant talks about the files, the strings and the second variables talks about how to do these kind of array kind of operations or C programming like structure for the for loop.

(Refer Slide Time: 15:08)



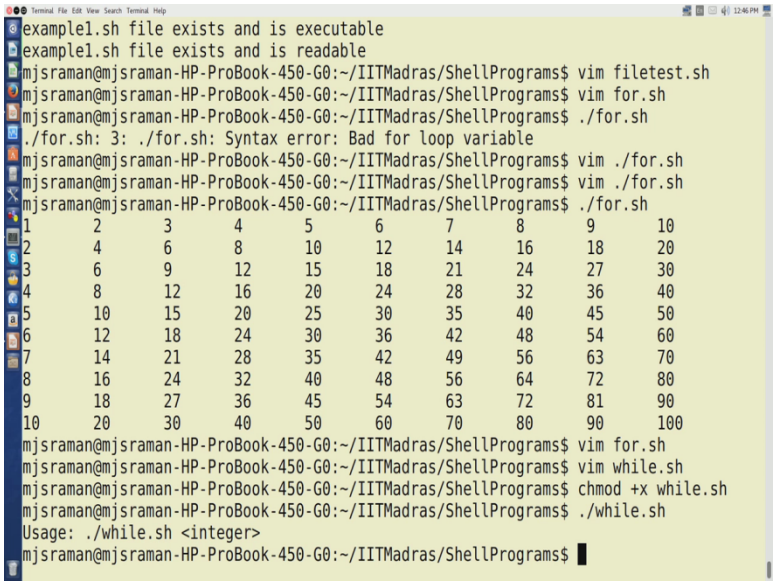
Now so we will move on so other than this for loop is there any other structure ok for loop control yes ok the shell also allows you to have a while construct. Ok! it checks for the true of false value of the condition before proceeding and the form of while loop is something like this; while and then it asks you for a condition and then you do bunch of commands and then finally you do it with a done statement. So let us try to take a look at the example and see how a while do done works.

(Refer Slide Time: 15:47)



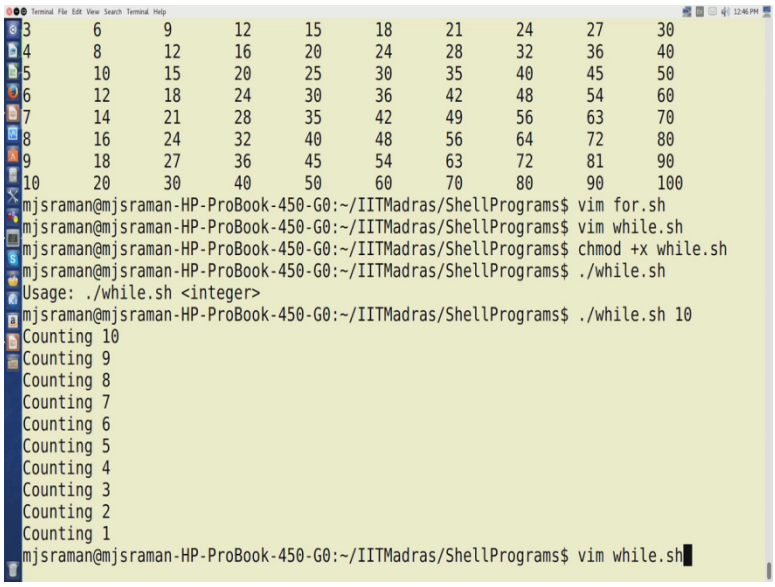
So let us take this program, so in this program what we do is the most important is line number 11. So in this program we try to find out whether the given number is greater than 0 so if the given number so if this condition that is number greater than 0 becomes true then it goes inside the loop ok and then executes whatever we want to execute ok and if this condition is false actually the loop exits. So if you look at this condition so let us what let us now try having understood that this while loop goes inside if this condition is true and then gets out of this the condition is false. Let us see what this program does. So we know that dollar 1 is the first parameter that you are going to take for the argument of the program, the first argument of the program. And that is being assigned to number and then while that number is greater than 0 we go on just printing that number and then reducing the number by 1, so essentially what happens is it acts like a counter which is always decrementing.

(Refer Slide Time: 17:09)



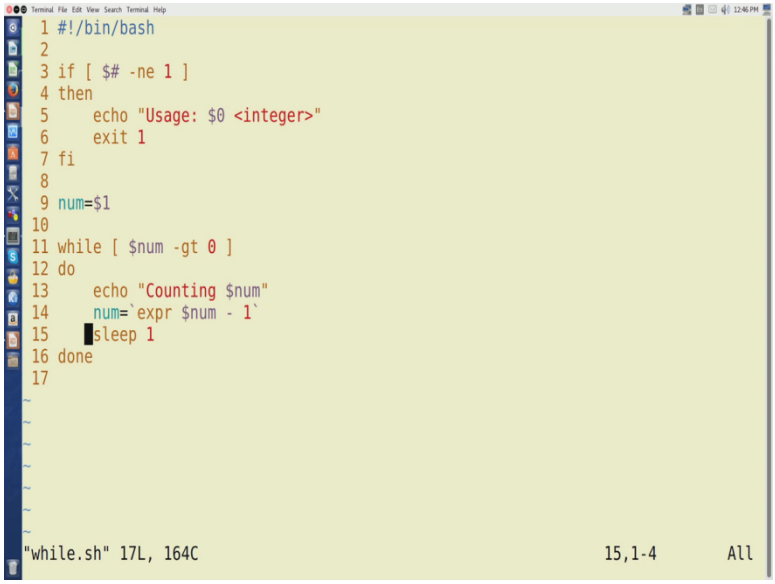
So let us see how to run this program we will change the execution mode and after this we will just run this program. So as I told you it tells you how to use this, so I have to give an integer number so let me say I give an integer number.

(Refer Slide Time: 17:20)



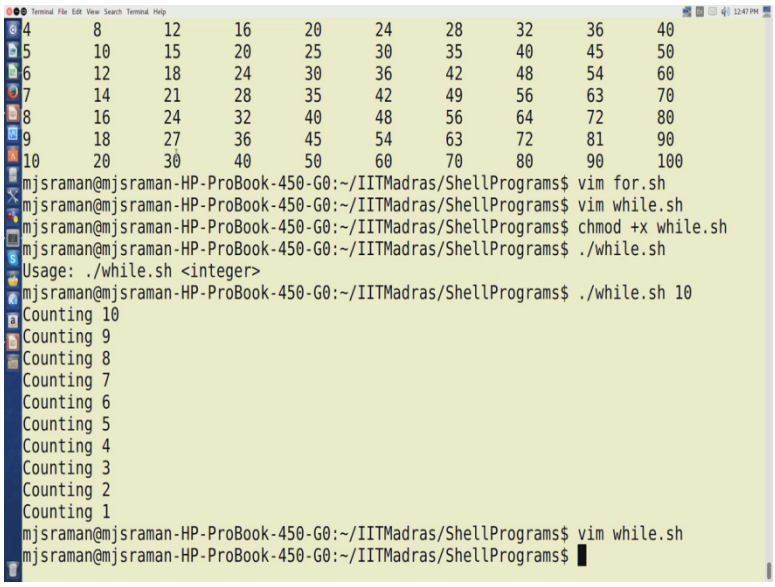
So let me say I give a integer of 10, so this starts counting and finally when it reaches so if you look at the condition the condition was that so let us look at the condition for this program. So if you look at the condition.

(Refer Slide Time: 17:48)



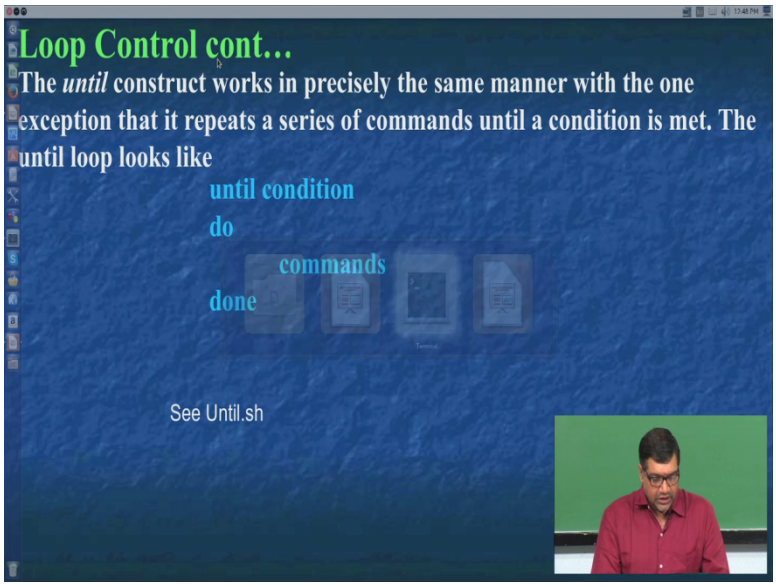
So condition was if the number is greater than 0 then go on counting if the number is equal to 0 then it will exit.

(Refer Slide Time: 17:55)



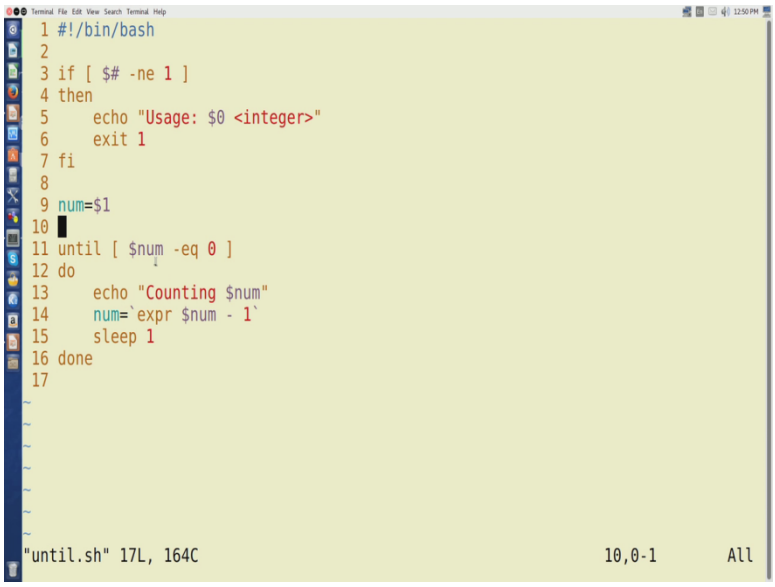
So as you can see in this program it counts upto 1 and after this 1 it exits the loop. Therefore this kind of while loop can be used while you want to find out before executing the loop you want to find whether some condition is true or not I mean if the condition is false then you can exit from the loop whereas in the for statement it will actually go into the condition and then it will initialize the value and then after that it checks for the condition so in this case the while loop essentially this difference between while and for loop, for loop we usually use it for kind of operating on files and things like that whereas this while loop you can use it for numbers, but there is no such strict rule as long as the input condition is Boolean you can operate on any of these loops. Either the for loop or the while loop.

(Refer Slide Time: 18:48)



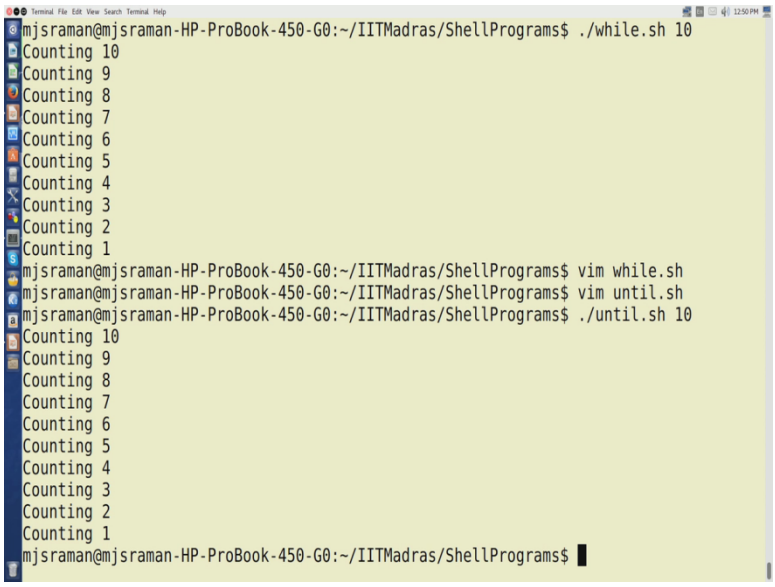
Shell programming also provides you a third mechanism for loop control ok this is known as until construct. The until construct works in precisely the same manner with the one exception that it repeats the series of command until the condition is met. So the idea behind this is that. Let us try to look at the syntax and then we will go to the condition and explanation of this. So it looks similar to the while loop so if you look at the previous slide we had while condition do commands and then done. And in this case you have until condition do commands. So in this case let us see whether the I mean if you can meet this particular condition ok? The loop will not exit. So let us take a look at this and then we will look at the explanation.

(Refer Slide Time: 19:36)



So in this case for example it is similar to the previous program and same counting number but look at the logic, so in the previous case what we found out was this number should be greater than as long as the number is greater than 0 usage go into the loop ok? So in this case what we are saying is until the number so if the number is equal to 0 ok then go out and exit. So that is what it says. So if you look at this if the number is not equal to 0 ok this loop will go on executing ok? So until the number is equal to 0 go on executing the loop if the number becomes 0 then exit. In the previous case if your remember what we have done was if you look at the while statement if the number is greater than 0 that is the condition what we had put actually this condition will become true if the number is greater than 0. Now in the previous in the until example this condition if the loop is executing then hopefully I mean the loop is executing the condition actually will become false and once this condition becomes false and when this condition is true then the loop will get out I mean off this, Ok? So if you look at this you can consider this as a inverse of a while loop or whatever it is, ok?

(Refer Slide Time: 20:55)



So this program does the same thing so if you give until dot sh ok? Then it counts starts 10 to 0. So I just wanted to show you that in shell programming there are many ways of getting a work done ok? It is not just one program will work correctly so it all depends how you input your creativity and write the code. So we will see more such examples I mean as we finish some 2 or 3 sections we will see some examples where it gets more and more complicated, but at least for a time being I hope you can understand that you can have for the loop control can be three mechanisms one you can use the for other you can use the while other you can do until ok? The logic for while and until is the inverse of each other whereas for the for loop you can have two variants the one variant looks at the string part of it the other variant looks similar to the C programming language part of it where you have the initial condition the final condition and you have the increment that happens.

So if you want you can use any of these techniques for example you can take as an exercise try to see whether you can print the same numbers reverse printing of reverse numbers using the for loop etc Thank You!