Information Security

Sri M J Shankar Raman,
Consultant of Computer Science and Engineering
Indian Institute of Technology Madras

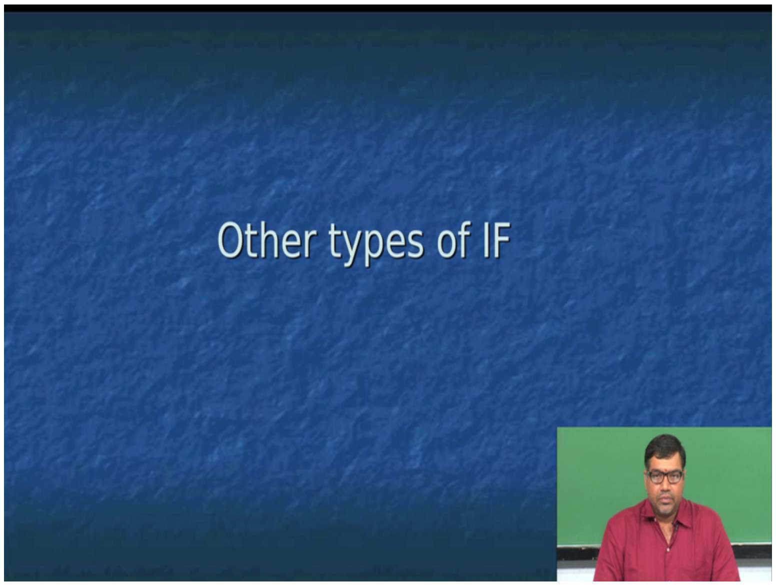
Module 34

Shell File Test

So in the last class we saw different types of ways to solve this problem of of finding the maximum of three numbers we also looked at how to handle functions in shell programming we also saw an example of a recursive function and we will continue with the same subject, what we will do this time is we will try to introduced a new variation of the statement.

So in the last class we saw different types of ways to solve this problem of of finding the maximum of three numbers we also looked at how to handle functions in shell programming we also saw an example of a recursive function and we will continue with the same subject what we will do this time is we will try to introduce a new variation of the if statement that in the else part also consists of another if if class so this is called as a else if class.

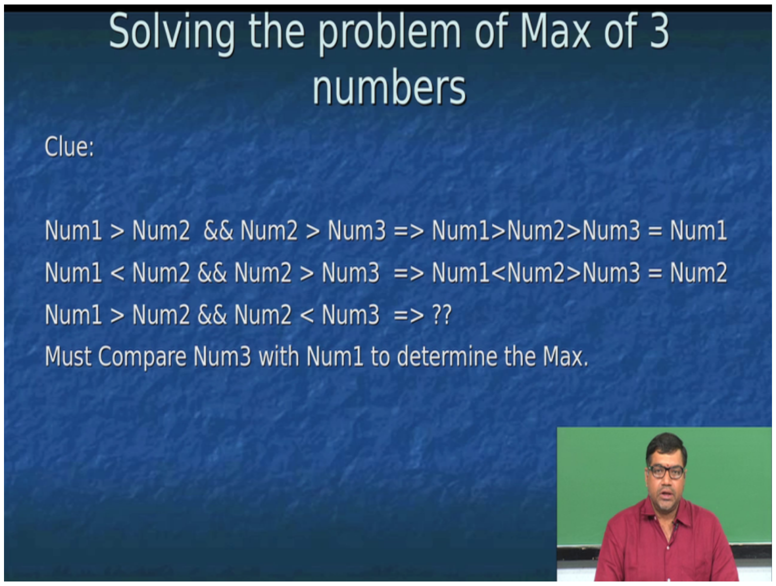
(Refer Slide Time: 00:50)



Other types of IF

So what we will do is in this case of solving the problem of trying to find out the maximum of three numbers. We found out that we had a logical error in the problem whereby in few of the cases we were not able to identify which was the maximum of three numbers.

(Refer Slide Time: 01:12)



Solving the problem of Max of 3 numbers

Clue:

Num1 > Num2 && Num2 > Num3 => Num1>Num2>Num3 = Num1

Num1 < Num2 && Num2 > Num3 => Num1<Num2>Num3 = Num2

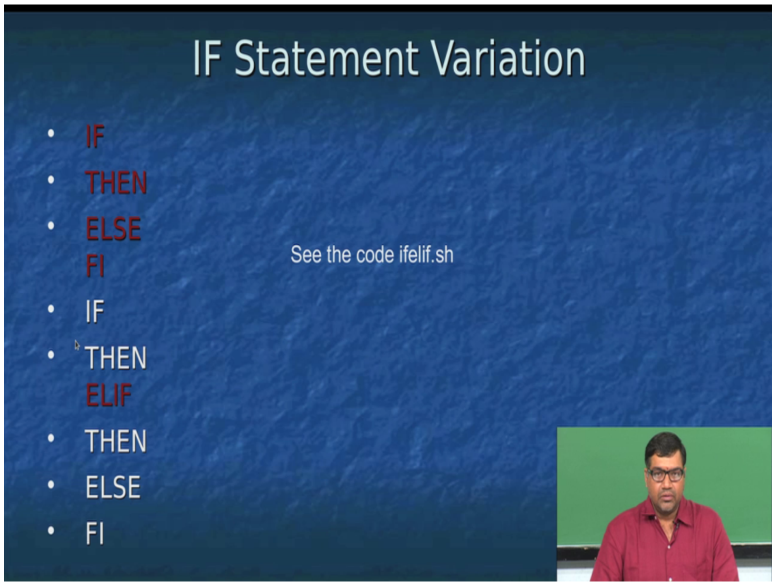
Num1 > Num2 && Num2 < Num3 => ??

Must Compare Num3 with Num1 to determine the Max.

Now how to now come up with logic for solving the problem. So if you look at this what we are what we will do is that in the if statement we have put statement like this if num1 is greater than num2 and and num 2 is greater than num3 then this imply that num1 is greater than num2 than num3 and therefore num1 was declared as the maximum of the three numbers.

Now we can proceed the logic in the same fashion and write the the statements that are given as given so in the first if condition we can announce that if num1 is greater than num2 and num 2 is greater than num3 then we announce that num1 is the greatest of the numbers and then we can exit the program and then we can again write another if statement where we say if num1 is less than num2 and num2 is greater than num3 then we can say that num2 is the greatest of the three numbers and this can continue like this will this will be one statement this is the another if statement and so on, so this is exactly what we tried last time.

(Refer Slide Time: 02:16)



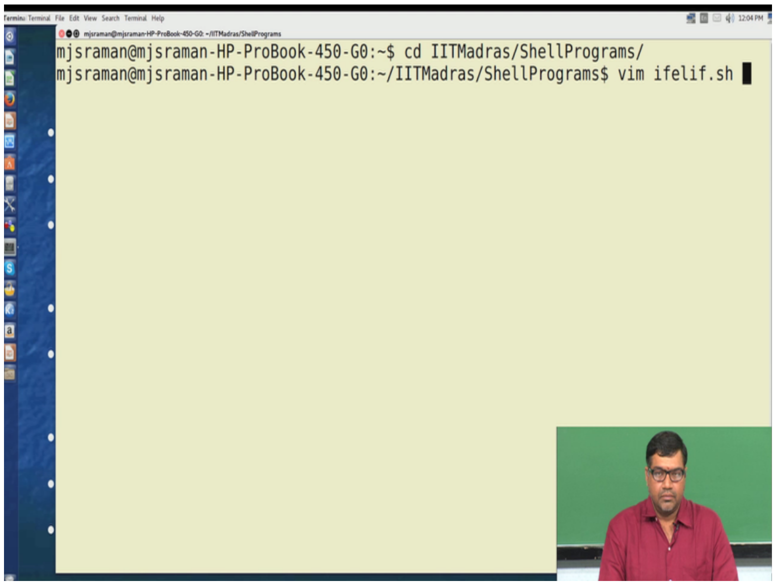
IF Statement Variation

- IF
- THEN
- ELSE
  FI
- IF
- THEN
  ELIF
- THEN
- ELSE
- FI

See the code ifelif.sh

What we will do now is we will try to introduce the if then and then if you'll look at this whatever is in red the else if part of this statement. So previously we solve the problem using if then else and fi and we found out that we had a problem with this but anyhow it was working for certain cases and we also understood the importance of doing unit testing and what we will do is we know come up with the new version of the code using this statement we will run some unit test on these and see whether the program is working fine.

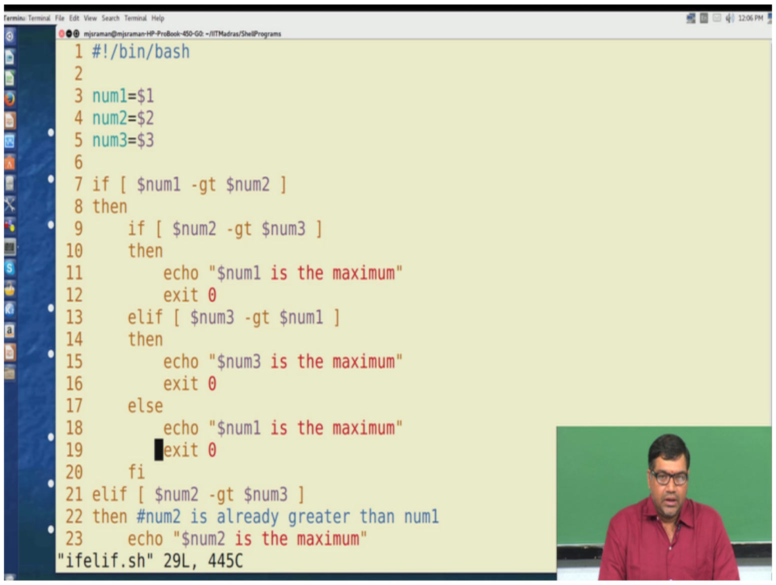
(Refer Slide Time: 02:55)



```
mjsraman@mjsraman-HP-ProBook-450-G0:~$ cd IITMadras/ShellPrograms/
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim ifelif.sh
```

So let us take the look at this program which is called ifelif dot.

(Refer Slide Time: 03:04)



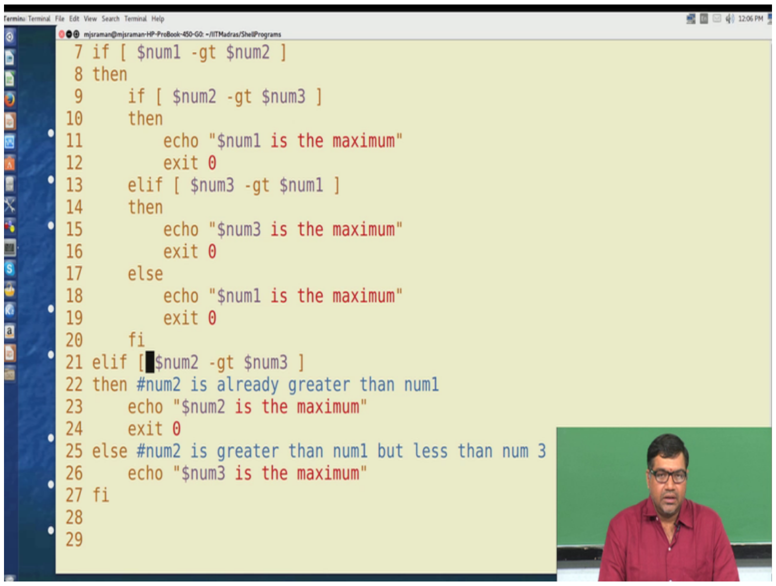
```
#!/bin/bash

num1=$1
num2=$2
num3=$3

if [ $num1 -gt $num2 ]
then
    if [ $num2 -gt $num3 ]
    then
        echo "$num1 is the maximum"
        exit 0
    elif [ $num3 -gt $num1 ]
    then
        echo "$num3 is the maximum"
        exit 0
    else
        echo "$num1 is the maximum"
        exit 0
    fi
elif [ $num2 -gt $num3 ]
then #num2 is already greater than num1
    echo "$num2 is the maximum"
"ifelif.sh" 29L, 445C
```

So in this program if you see we are now taking the three numbers which are the parameters as dollar 1 dollar 2 dollar 3 and what we are doing is we are actually executing inside the if statement ok if you look at this now we are trying to find out number 1 is greater than number 2 and if this number 1 is greater than number 2 then we compare number2 whether number 2 is greater than number 3.

So if number 2 is greater than number 3. then we announce that number1 is the maximum which is according to a logic which will be shown in the previous line now then Then we get out of this and the most important part occurs here look at this if number 2 is not greater than number 3 that means number 2 is less than number 3 then we again try to see if number 3 is greater than number 1 ok because what happens is this number 1 is definitely greater than number 2 but we see that number 3 is also greater than number2 .

Therefore we've to find in whether number 3 and number 1 the relationship between number 3 and number 1. So if number three is greater, then we announce the result to be number 3 is the maximum else we announce number 1 is the maximum and if you look at this part ok so the if you look at this this if then else if else feet now this becomes one complete if then else statement ok so if you look at this this is a part of so inside the else part we are having another if then else ok and if so now this else part ok after this the else part consists of a if statement and this if statement consists of two else parts ok.

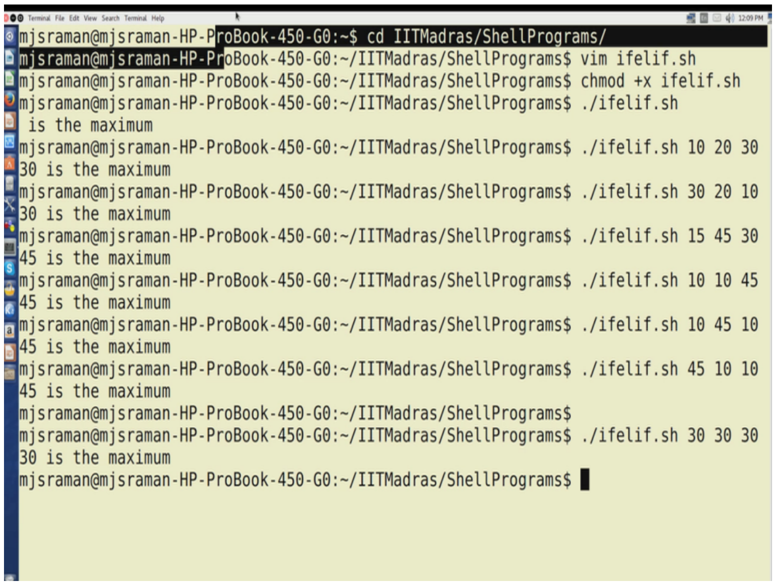
(Refer Slide Time: 04:46)



```
if [ $num1 -gt $num2 ]
then
    if [ $num2 -gt $num3 ]
    then
        echo "$num1 is the maximum"
        exit 0
    elif [ $num3 -gt $num1 ]
    then
        echo "$num3 is the maximum"
        exit 0
    else
        echo "$num1 is the maximum"
        exit 0
    fi
elif [ $num2 -gt $num3 ]
then #num2 is already greater than num1
    echo "$num2 is the maximum"
    exit 0
else #num2 is greater than num1 but less than num 3
    echo "$num3 is the maximum"
fi
```

And then here what happens is now we are taking the second condition if the number 2 is greater than number 3. So if number 2 is greater than number 3 then we say that number 2 is already number 2 is the maximum and so on.

(Refer Slide Time: 05:00)



```
mjsraman@mjsraman-HP-ProBook-450-G0:~$ cd IITMadras/ShellPrograms/
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim ifelif.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ chmod +x ifelif.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh
 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 10 20 30
30 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 30 20 10
30 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 15 45 30
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 10 10 45
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 10 45 10
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 45 10 10
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 30 30 30
30 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ 
```

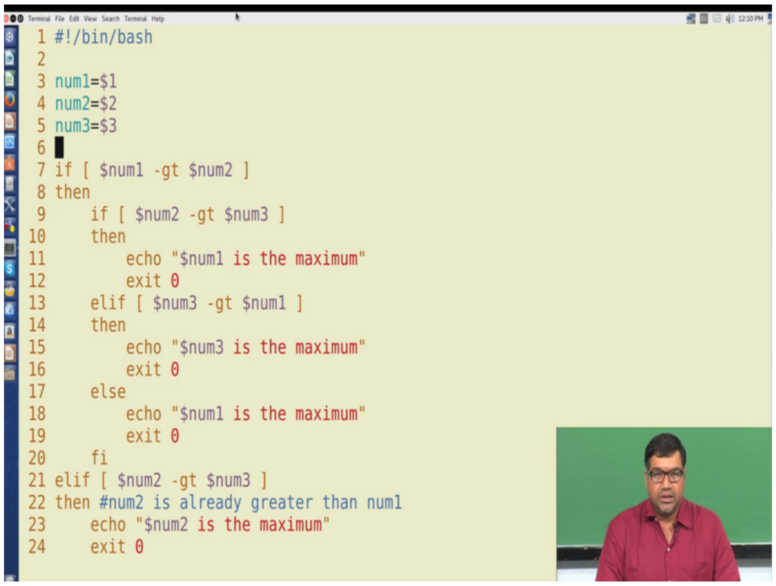
So let us see whether this program this logic whatever you've found will give the right answer. So we'll run this program and then we'll see whether this gives the right answer. So what we do is we save this program and we'll run this. Since we, let's change the executable permission for this, and then we can run this program. So if we don't given any parameters, now the program, actually there is a bug in this program that we are not doing the usage take. So if we are listen to the lectures earlier we told you that whenever you write such programs you should have the usage itself. Now that we've written the program we know the usage so there's no problem here. But in any case you should always write how to use the program.

So in this case let's try to give the, we've to give an input of three numbers so we'll give 10 20 and 30 and this exactly prints that 30 is the maximum. Now let us try to find out ok what happens if we reverse the order ok. So we'll give 30 20 and 10. So this program again prints 30 is the maximum. Let's give some other number to to, so let's say let's give15, 45 and then 30 and it prints 45 is the maximum.

So the first three test cases where we give the first number as the, the third number as the maximum, the first number as the maximum and the second number as the maximum this program passes. Now what we'll try to do is in the previous program if you had the if statement that we saw, we saw that when we gave two numbers as equal it was not trying to detect the maximum of the numbers. Therefore what we do is we'll try to give 10 10 and then 45 so elseif.sh 10 10 and 45 ok now it clearly detects that 45 is the maximum number. Now we'll give this position this 45 in between these two ok and then see ok then also it works correctly now we’ll position this 45 as a first one and then we’ll 10 and 10 and even in this case it detects the the maximum number.

Now if you remember this was one of the cases where we had an error in the previous program, now that it has detected the maximum number, so what we’ll try to do is we’ll try to give other test cases what we try to give a say all the numbers as equal ok. So if we give all the numbers is equal say 30 30 and 30 and it says 30 is the maximum which is ok. Now the question is, is this program working correctly? Ok we will leave this as an exercise to you to show to prove or disprove whether this program is correct.

(Refer Slide Time: 07:50)

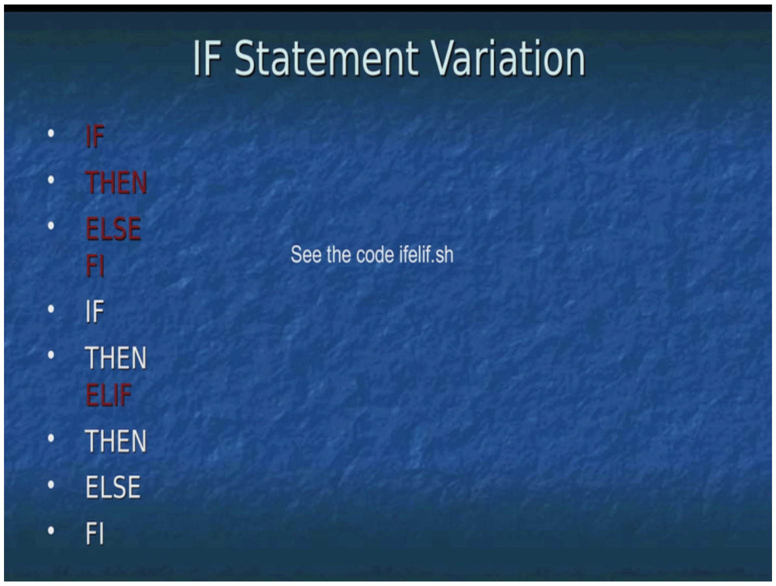


```
#!/bin/bash

num1=$1
num2=$2
num3=$3

if [ $num1 -gt $num2 ]
then
    if [ $num2 -gt $num3 ]
    then
        echo "$num1 is the maximum"
        exit 0
    elif [ $num3 -gt $num1 ]
    then
        echo "$num3 is the maximum"
        exit 0
    else
        echo "$num1 is the maximum"
        exit 0
    fi
elif [ $num2 -gt $num3 ]
then #num2 is already greater than num1
    echo "$num2 is the maximum"
    exit 0
```

So what we will do is we will now again see the program once probably you can take a look at this or note down the logic and you can prove using say dish and tree or some other technique, dish and table or whatever technique to find out whether this program works correctly or not. We saw that it passed the six text cases but that doesn't mean that there may not be any errors in the program.

So if you really want to prove that the program works correctly probably you might have to move for some kind of formal proof mechanism but at least in your case you can try to make use of dish and tables please refer to the web1 what is a dish and table or a dish and tree and see whether this program works correctly.

(Refer Slide Time: 08:33)



IF Statement Variation

- IF
- THEN
- ELSE
  FI
- IF
- THEN
  ELIF
- THEN
- ELSE
- FI

See the code ifelif.sh

Now moving on so what we do is we we we see that ok the if statement until now we have been only doing the comparison operators with numbers, now what we will try to do is we will try to see whether we can do a file check operations.

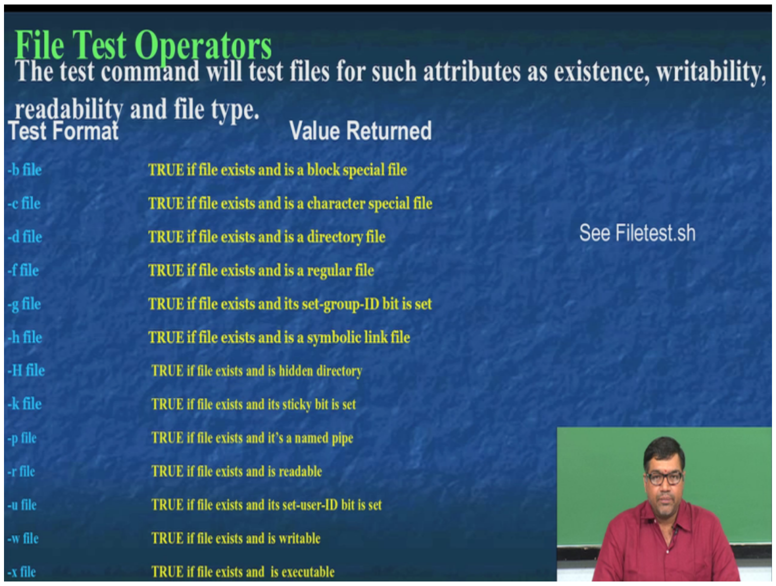
(Refer Slide Time: 08:53)

File Test Operators

See Filetest.sh

So one of the things with shell scripting is that it is not necessary that the the the conditions that is inside the if statement must be only numeric as long as the condition is boolean ok the if statement will work.

(Refer Slide Time: 09:08)



File Test Operators

The test command will test files for such attributes as existence, writability, readability and file type.

Test Format	Value Returned
-b file	TRUE if file exists and is a block special file
-c file	TRUE if file exists and is a character special file
-d file	TRUE if file exists and is a directory file
-f file	TRUE if file exists and is a regular file
-g file	TRUE if file exists and its set-group-ID bit is set
-h file	TRUE if file exists and is a symbolic link file
-H file	TRUE if file exists and is hidden directory
-k file	TRUE if file exists and its sticky bit is set
-p file	TRUE if file exists and it's a named pipe
-r file	TRUE if file exists and is readable
-u file	TRUE if file exists and its set-user-ID bit is set
-w file	TRUE if file exists and is writable
-x file	TRUE if file exists and is executable

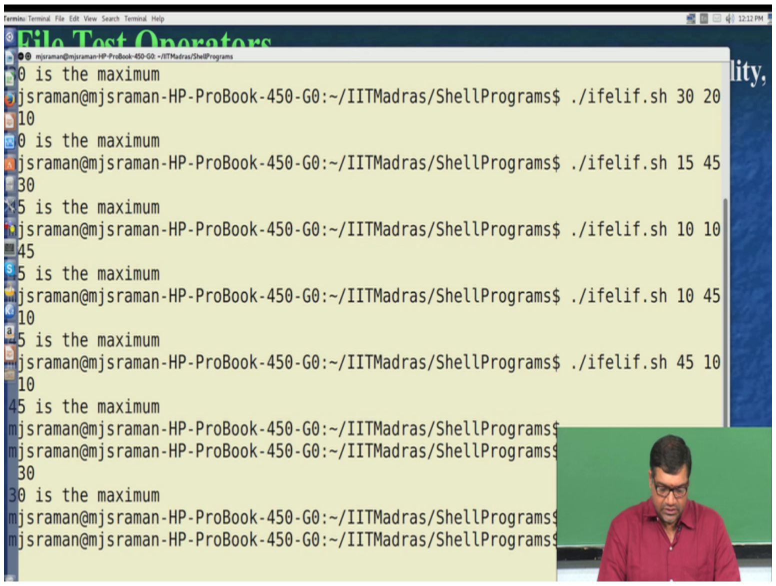
See Filetest.sh

So what what, I mean most of cases in shell scripting you may not be playing with numbers but you'll be playing with either files or strings ok, of course in the forth coming classes we'll see how to handle strings within the statement or within the conditional part of the if statement. Currently what we'll try to do is we will try to understand a boot file test operators which can be used inside the if statement ok.

The idea behind the file test operators is that since shell scripting is going to handle more of your operating system files and directories, they are provided special facilities to test the whether there is the your file whatever file you are giving is existing or not and whether your file is a special file and provided and whether the file has got read permission or write permission or execute permission and so on, and you can also find out what type of file it is.

So if you look at the way you are suppose to give of identify what type of file it is, we will now look at a example and then come back to the slide again and see how the example and the slide matches.

(Refer Slide Time: 10:23)

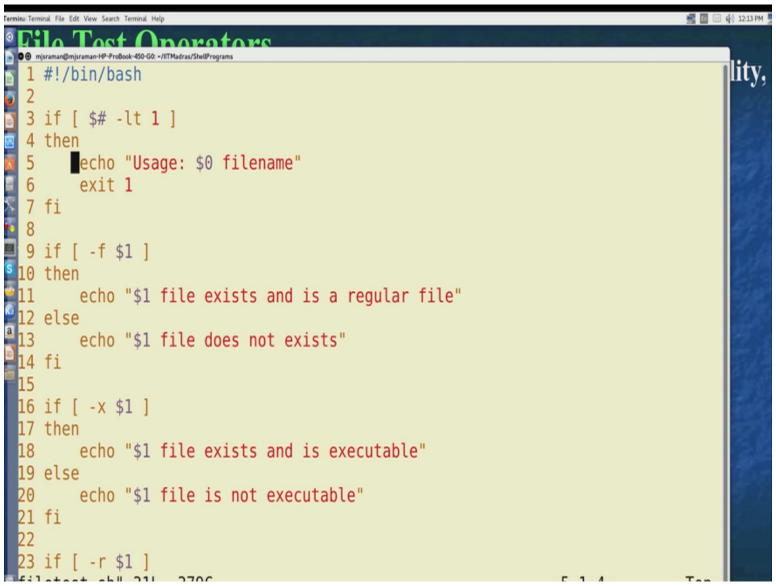


File Test Operators

```
0 is the maximum
jsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 30 20
10
0 is the maximum
jsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 15 45
30
5 is the maximum
jsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 10 10
45
5 is the maximum
jsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 10 45
10
5 is the maximum
jsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 45 10
10
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
30
30 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
```

So what we will do is we'll try to take this file called file test dot sh.

(Refer Slide Time: 10:32)



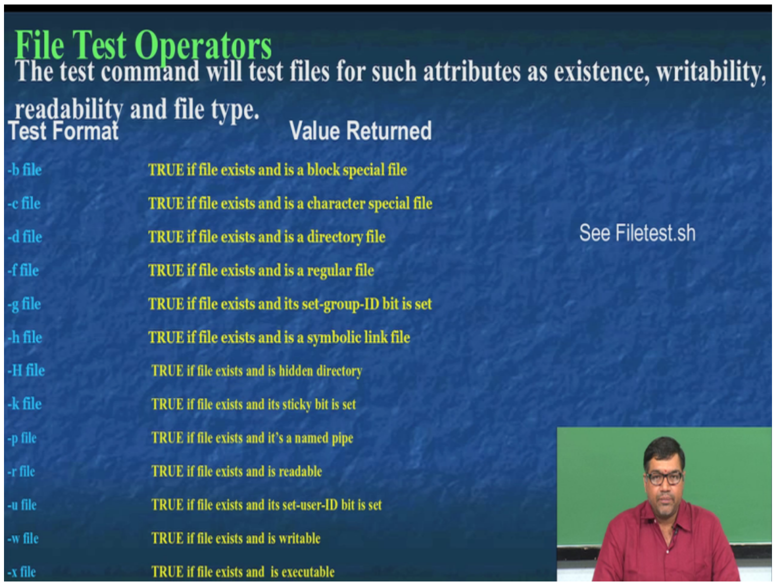
File Test Operators

```
#!/bin/bash

if [ $# -lt 1 ]
then
    echo "Usage: $0 filename"
    exit 1
fi

if [ -f $1 ]
then
    echo "$1 file exists and is a regular file"
else
    echo "$1 file does not exists"
fi

if [ -x $1 ]
then
    echo "$1 file exists and is executable"
else
    echo "$1 file is not executable"
fi

if [ -r $1 ]
```

Let us see how to do this operation ok, so let us take the statement ok so this program actually expects a file name to be given to it ok. So since it expects a file name to be given to it, what we do is the file name will come as the parameter dollar1, now what we are trying to do in this program is we have put minus f , see if you look at the statement it is I f space then we are opening the square bracket then we are giving a blank we are giving a minus f ok and then space and then we are putting dollar 1, dollar 1 you know is the first parameter I know that is nothing but the file name that you'll be supplying and there's a space and then you are closing the brace.

So if I use this condition say minus f dollar 1 or minus f file name, this condition in shell script I tries to identify whether this file exists and is a regular file, regular file is any text file or whatever it is ok, and there are regular files and special files hope you've heard about that in your previous lectures, so this is whether it's a regular file. So if the statement becomes true that means if minus f dollar 1 becomes true then this program prints dollar 1 file exists and is a regular file otherwise it prints the file does not exists.

(Refer Slide Time: 11:58)



File Test Operators

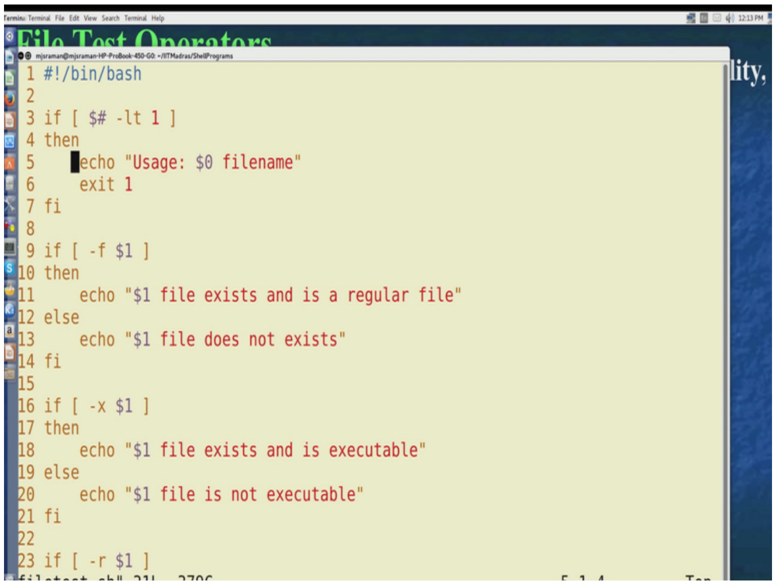
The test command will test files for such attributes as existence, writability, readability and file type.

Test Format	Value Returned
-b file	TRUE if file exists and is a block special file
-c file	TRUE if file exists and is a character special file
-d file	TRUE if file exists and is a directory file
-f file	TRUE if file exists and is a regular file
-g file	TRUE if file exists and its set-group-ID bit is set
-h file	TRUE if file exists and is a symbolic link file
-H file	TRUE if file exists and is hidden directory
-k file	TRUE if file exists and its sticky bit is set
-p file	TRUE if file exists and it's a named pipe
-r file	TRUE if file exists and is readable
-u file	TRUE if file exists and its set-user-ID bit is set
-w file	TRUE if file exists and is writable
-x file	TRUE if file exists and is executable

See Filetest.sh

Now let's go back to the slide and see this. So what we have used is this part of the code so minus f file so it says true if the file exists and is a regular file.

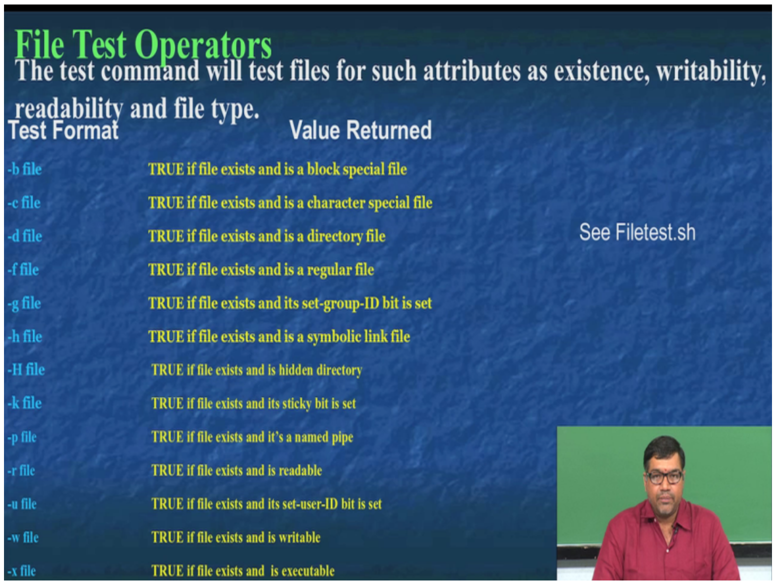
(Refer Slide Time: 12:15)



```
#!/bin/bash

if [ $# -lt 1 ]
then
    echo "Usage: $0 filename"
    exit 1
fi

if [ -f $1 ]
then
    echo "$1 file exists and is a regular file"
else
    echo "$1 file does not exists"
fi

if [ -x $1 ]
then
    echo "$1 file exists and is executable"
else
    echo "$1 file is not executable"
fi

if [ -r $1 ]
```

So similarly what we can do is in order to check whether the file have got read permission, whether it has got write permission or whether it has got executable permission and so on, what we can do is we can perform, checks like this so if you look at line number 16 we see that we are checking using an option called minus x.

(Refer Slide Time: 12:34)



File Test Operators

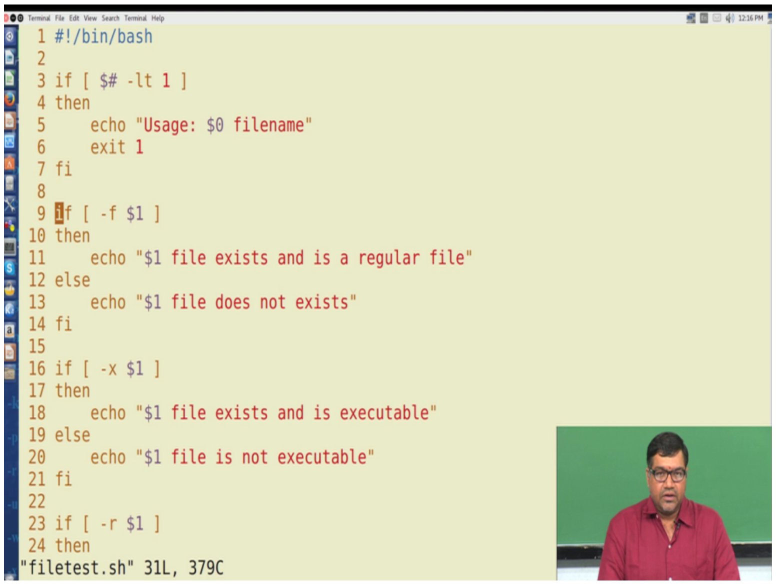
The test command will test files for such attributes as existence, writability, readability and file type.

Test Format	Value Returned
-b file	TRUE if file exists and is a block special file
-c file	TRUE if file exists and is a character special file
-d file	TRUE if file exists and is a directory file
-f file	TRUE if file exists and is a regular file
-g file	TRUE if file exists and its set-group-ID bit is set
-h file	TRUE if file exists and is a symbolic link file
-H file	TRUE if file exists and is hidden directory
-k file	TRUE if file exists and its sticky bit is set
-p file	TRUE if file exists and it's a named pipe
-r file	TRUE if file exists and is readable
-u file	TRUE if file exists and its set-user-ID bit is set
-w file	TRUE if file exists and is writable
-x file	TRUE if file exists and is executable

See Filetest.sh

So let's go and see what this minus x option is about, so if you will look at this minus x option which is the last one that is shown here it tells you that it is true if the file exist and it is an executable.

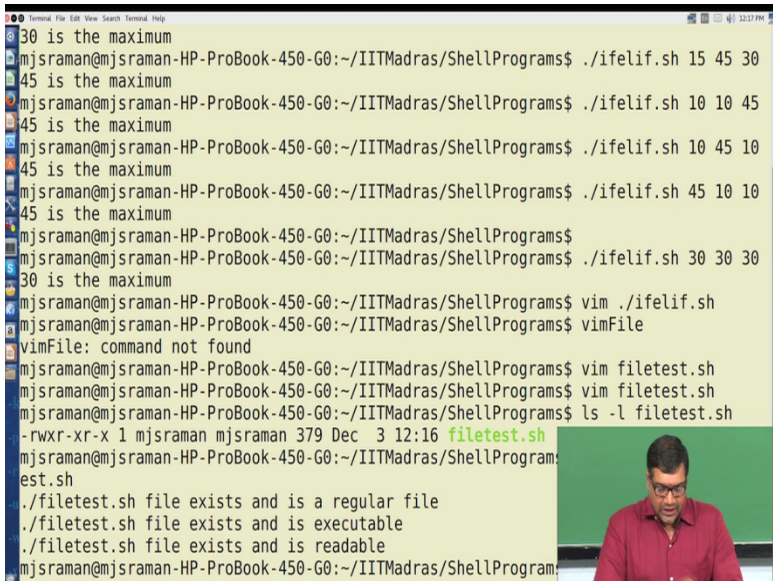
(Refer Slide Time: 12:50)



```
 1 #!/bin/bash
 2
 3 if [ $# -lt 1 ]
 4 then
 5     echo "Usage: $0 filename"
 6     exit 1
 7 fi
 8
 9 if [ -f $1 ]
10 then
11     echo "$1 file exists and is a regular file"
12 else
13     echo "$1 file does not exists"
14 fi
15
16 if [ -x $1 ]
17 then
18     echo "$1 file exists and is executable"
19 else
20     echo "$1 file is not executable"
21 fi
22
23 if [ -r $1 ]
24 then
"filetest.sh" 31L, 379C
```

So first what we do is we just invoke the bash shell and then what we try to do is we if the program is apply one parameter ok then the argument, parameter of the argument, then it will work correctly if you don't supply the parameter then what happens is that it just goes down ok, of course this program has a error you need to find out what error it is ok in the usage a part of it ok.

(Refer Slide Time: 13:20)

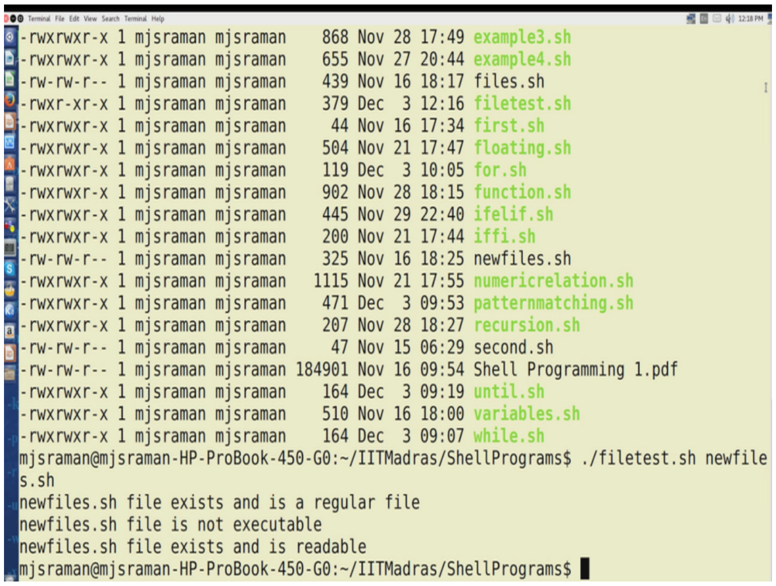


```
30 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 15 45 30
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 10 10 45
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 10 45 10
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 45 10 10
45 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./ifelif.sh 30 30 30
30 is the maximum
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim ./ifelif.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vimFile
vimFile: command not found
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim filetest.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim filetest.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ls -l filetest.sh
-rwxr-xr-x 1 mjsraman mjsraman 379 Dec  3 12:16 filetest.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellProgram
est.sh
./filetest.sh file exists and is a regular file
./filetest.sh file exists and is executable
./filetest.sh file exists and is readable
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellProgram
```

And then finally if everything is fine if you provide one argument or one file then it tries to identify minus x option and minus r option and so on so minus r option is to find out whether the file is readable or not. So let us see how this program works and we will try to show with an example, so let us take one file so probably let's say file test the current file itself so we will test the current file itself and if you'll look at the current file it's a regular file because it's a shell file.

So if you look at the options, at the bottom, I mean it tells you that it has read, write and executable permissions for the current user. So what we will try to do is we will try to execute this file ok file test dot sh and then we we'll do the file the same file name, so what we are testing our own file using this program. so it tells you that it's a regular file and it's an executable, it's a readable file and it's a regular file as well as it's an executable file.

(Refer Slide Time: 14:22)

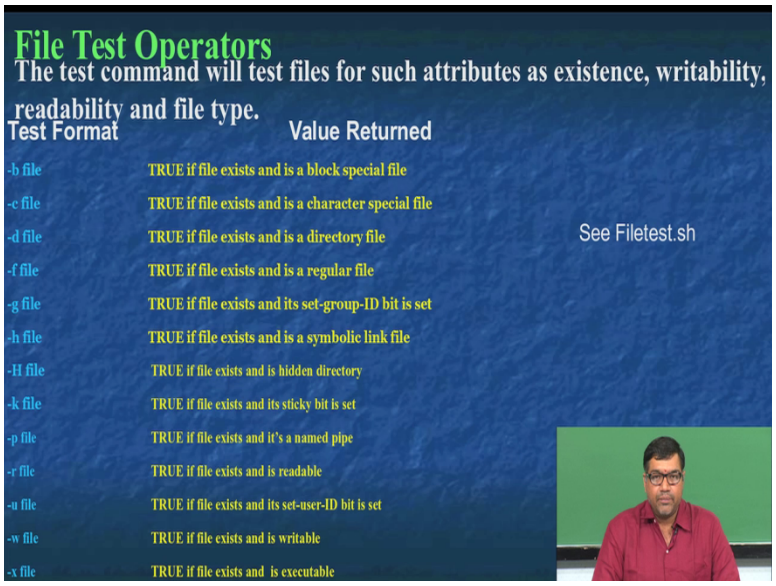


```
-rwxrwxr-x 1 mjsraman mjsraman    868 Nov 28 17:49 example3.sh
-rwxrwxr-x 1 mjsraman mjsraman    655 Nov 27 20:44 example4.sh
-rw-rw-r-- 1 mjsraman mjsraman    439 Nov 16 18:17 files.sh
-rwxr-xr-x 1 mjsraman mjsraman    379 Dec  3 12:16 filetest.sh
-rwxrwxr-x 1 mjsraman mjsraman     44 Nov 16 17:34 first.sh
-rwxrwxr-x 1 mjsraman mjsraman    504 Nov 21 17:47 floating.sh
-rwxrwxr-x 1 mjsraman mjsraman    119 Dec  3 10:05 for.sh
-rwxrwxr-x 1 mjsraman mjsraman    902 Nov 28 18:15 function.sh
-rwxrwxr-x 1 mjsraman mjsraman    445 Nov 29 22:40 ifelif.sh
-rwxrwxr-x 1 mjsraman mjsraman    200 Nov 21 17:44 iffi.sh
-rw-rw-r-- 1 mjsraman mjsraman    325 Nov 16 18:25 newfiles.sh
-rwxrwxr-x 1 mjsraman mjsraman   1115 Nov 21 17:55 numericrelation.sh
-rwxrwxr-x 1 mjsraman mjsraman    471 Dec  3 09:53 patternmatching.sh
-rwxrwxr-x 1 mjsraman mjsraman    207 Nov 28 18:27 recursion.sh
-rw-rw-r-- 1 mjsraman mjsraman     47 Nov 15 06:29 second.sh
-rw-rw-r-- 1 mjsraman mjsraman 184901 Nov 16 09:54 Shell Programming 1.pdf
-rwxrwxr-x 1 mjsraman mjsraman    164 Dec  3 09:19 until.sh
-rwxrwxr-x 1 mjsraman mjsraman    510 Nov 16 18:00 variables.sh
-rwxrwxr-x 1 mjsraman mjsraman    164 Dec  3 09:07 while.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./filetest.sh newfile
s.sh
newfiles.sh file exists and is a regular file
newfiles.sh file is not executable
newfiles.sh file exists and is readable
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ 
```

Now, so let’s now go ahead and try to see what happens if I give a program which is not a executable file, so there are, there is a file called new files dot sh so we’ll run the same program on new files dot sh so what happen, let’s see what happens when you run the same program.

So if you look at this it says that even though it’s a shell program and even though it is readable now what has happened is that since the file is not executable and this can be seen by looking at the fact that the new files dot sh the x part of it for both the user group and others it seems to be dash ok. So in this way if you want to identify now this identify what happens with this file, file permissions you can use the minus f file name or you can use any of these options that we had discussed.

(Refer Slide Time: 15:10)



File Test Operators

The test command will test files for such attributes as existence, writability, readability and file type.

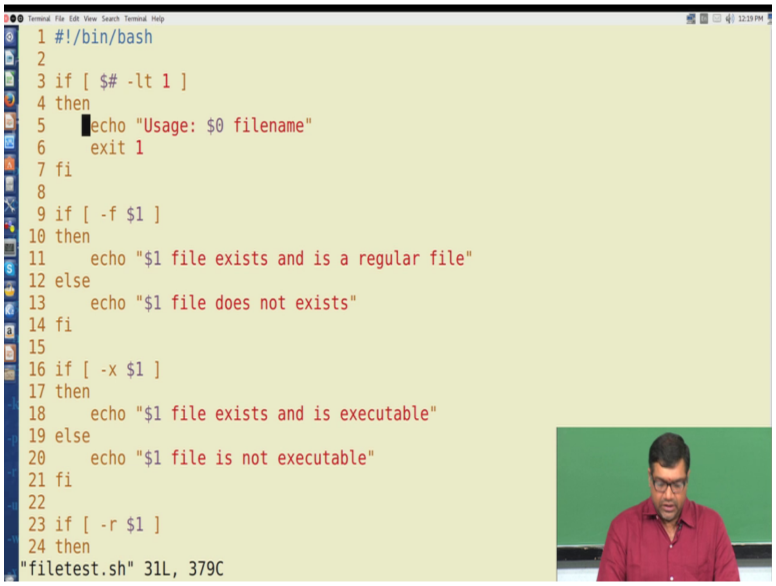
Test Format	Value Returned
-b file	TRUE if file exists and is a block special file
-c file	TRUE if file exists and is a character special file
-d file	TRUE if file exists and is a directory file
-f file	TRUE if file exists and is a regular file
-g file	TRUE if file exists and its set-group-ID bit is set
-h file	TRUE if file exists and is a symbolic link file
-H file	TRUE if file exists and is hidden directory
-k file	TRUE if file exists and its sticky bit is set
-p file	TRUE if file exists and it's a named pipe
-r file	TRUE if file exists and is readable
-u file	TRUE if file exists and its set-user-ID bit is set
-w file	TRUE if file exists and is writable
-x file	TRUE if file exists and is executable

See Filetest.sh

So you have options like minus b where it says it's a block file I mean , then you have a character file, I mean this block file and character file are used for in case of device drivers then you want to find out whether it's a directory now you can use the minus d option then you can have options like minus g minus h and so on.

And all these things you can use when you come across different types of scenarios where you want to find out what type of file it is whether the sticky bit has been set or whether it's a pipe file I mean you you would have heard about named pipes where you create a file which through which communication happens, so all these things you can find out using this file test operators. Now one of the things that we should also note with these kinds of file test operators is that if you look at this code.

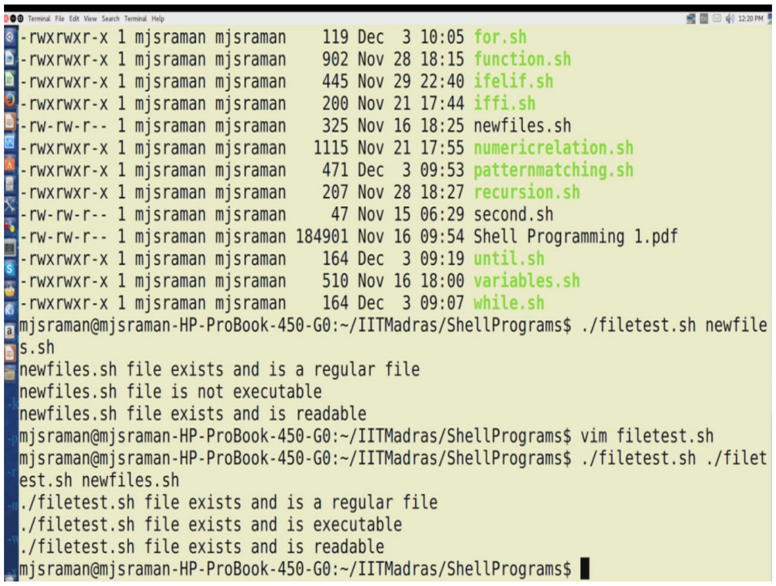
(Refer Slide Time: 16:09)



```
#!/bin/bash

if [ $# -lt 1 ]
then
    echo "Usage: $0 filename"
    exit 1
fi

if [ -f $1 ]
then
    echo "$1 file exists and is a regular file"
else
    echo "$1 file does not exists"
fi

if [ -x $1 ]
then
    echo "$1 file exists and is executable"
else
    echo "$1 file is not executable"
fi

if [ -r $1 ]
then
"filetest.sh" 31L, 379C
```

So previously we got an error for about usage so if you, the most important part in any shell script is to tell the user how to use this, so if you look at this first part one of the bugs that I've mentioned is that if I give more than file one file name what happens to this code, ok and probably it will throughout an error, so let's try to find out whether such a thing happens.

(Refer Slide Time: 16:26)



```
-rwxrwxr-x 1 mjsraman mjsraman    119 Dec  3 10:05 for.sh
-rwxrwxr-x 1 mjsraman mjsraman    902 Nov 28 18:15 function.sh
-rwxrwxr-x 1 mjsraman mjsraman    445 Nov 29 22:40 ifelif.sh
-rwxrwxr-x 1 mjsraman mjsraman    200 Nov 21 17:44 iffi.sh
-rw-rw-r-- 1 mjsraman mjsraman    325 Nov 16 18:25 newfiles.sh
-rwxrwxr-x 1 mjsraman mjsraman   1115 Nov 21 17:55 numericrelation.sh
-rwxrwxr-x 1 mjsraman mjsraman    471 Dec  3 09:53 patternmatching.sh
-rwxrwxr-x 1 mjsraman mjsraman    207 Nov 28 18:27 recursion.sh
-rw-rw-r-- 1 mjsraman mjsraman     47 Nov 15 06:29 second.sh
-rw-rw-r-- 1 mjsraman mjsraman 184901 Nov 16 09:54 Shell Programming 1.pdf
-rwxrwxr-x 1 mjsraman mjsraman    164 Dec  3 09:19 until.sh
-rwxrwxr-x 1 mjsraman mjsraman    510 Nov 16 18:00 variables.sh
-rwxrwxr-x 1 mjsraman mjsraman    164 Dec  3 09:07 while.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./filetest.sh newfile
s.sh
newfiles.sh file exists and is a regular file
newfiles.sh file is not executable
newfiles.sh file exists and is readable
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim filetest.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./filetest.sh ./filet
est.sh newfiles.sh
./filetest.sh file exists and is a regular file
./filetest.sh file exists and is executable
./filetest.sh file exists and is readable
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
```

So if you, for example if I gave file test dot sh then I give some two files, let’s say file test dot sh, so file test dot sh and as well as new files dot sh, now you see what happens is it doesn’t test the second program, I mean it test only the first program it doesn’t test the second program.

So your script ok whatever you write as I told you when you do unit test, you will also identify these type of errors, so in this case what happens is that you’ve to repeat this again and again to find out what type of files and all that, so the question is does shell provides a mechanism by which I can repeatedly execute the same command on different test set of files. If you would have heard about the xargs in the previous sections ok when you were taught about UNIX commands but the functionality of xargs can also be brought by using shell script.

And we will see in the next section how we can use, get this using the for statement ok and so in in the for statement what we could do is that we can repeatedly execute the current program with every file and let us see in the next section how to do about doing it,

Thank you.