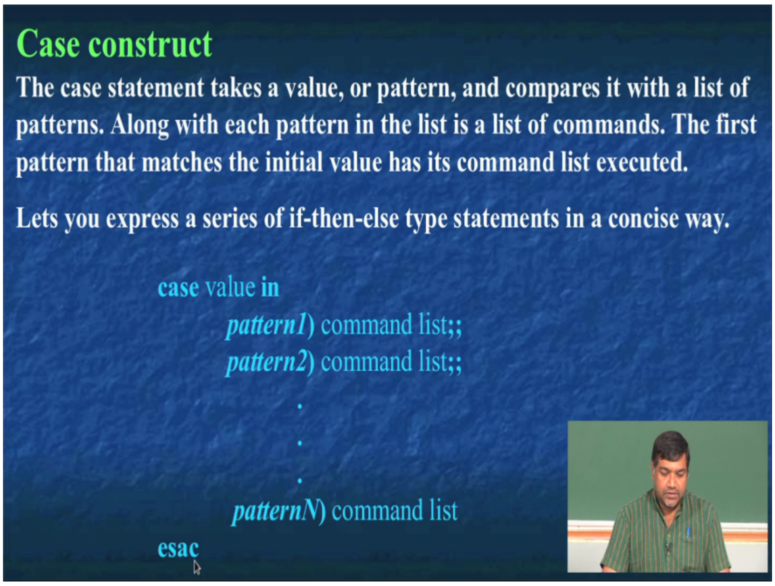
Information Security 3
Sri M J Shankar Raman,
Consultant Department of Computer Science and Engineering,
Indian Institute of Technology Madras
Module 38
Shell Case Statement

Hi, Hello there! Welcome to this session on shell programming where we are going to take a look at a new Construct called the case statement.

(Refer Slide Time: 00:30)



Case construct

The case statement takes a value, or pattern, and compares it with a list of patterns. Along with each pattern in the list is a list of commands. The first pattern that matches the initial value has its command list executed.

Lets you express a series of if-then-else type statements in a concise way.

```
case value in
        pattern1) command list;;
        pattern2) command list;;
                .
                .
                .
        patternN) command list
esac
```

The idea behind a case construct is that we had seen in the previous classes that we had used if else construct. so what we were trying to do with if else construct was there we were trying to take multiple decisions. Say for examples we found out that if condition a is happened then I do not want any of the condition b, c and d to happen but I want to continue with the flow of the program. So I am given multiple choices or multiple paths and out of those multiple paths I have to choose one path, so that is the reason why I used an if else statement.

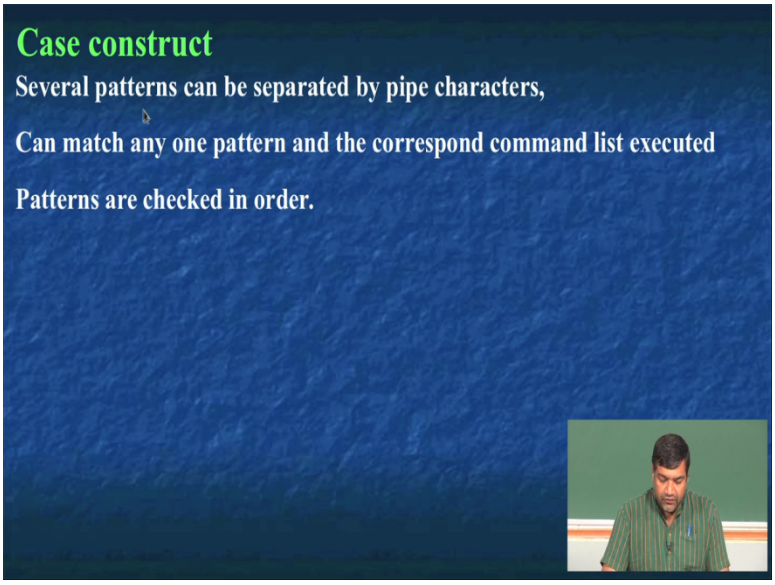
Now sometimes it can so happen that if else statement can become pretty complex ok that means choosing one path out of the multiple paths can be coded in a very complex fashion. So in order to ease out that problem the shell allows you to have something mosaic case construct and it is essentially if there is a series of if then else type of statements then this case construct allows you to have very nice way of expressing your logic.

So that is one of the reasons where we use the case construct. What happens with case construct is that it takes a value or a pattern and then compares it with a list of pre decided patterns that you give in your program and out of these patterns if one of the patterns matches then it takes the path. So I know this is a kind of an abstract way of explaining a case construct.

So essentially in a form of syntax we have the following ok? So in the form of syntax what we see is that we have a case ok And then we have this is the particular value that user gives and each of these patterns refers to one particular path that you want to take out of this whole logic you know if you look at this statement ok I take I this I am passing a value and this value could match one of these patterns and if it matches let us say pattern 2 then all these commands that are given upto this double semi colon will get executed suppose the value matches pattern 1 then all these commands here upto this double semi colon will be used.

Many of you who are familiar with C programming language will understand this is equal to something like switch case statement and one of the things that we do in shell as usual is that once I start the case I have to end with the reverse of the case which is esac ok. So this forms a complete block.

(Refer Slide Time: 03:48)



Case construct

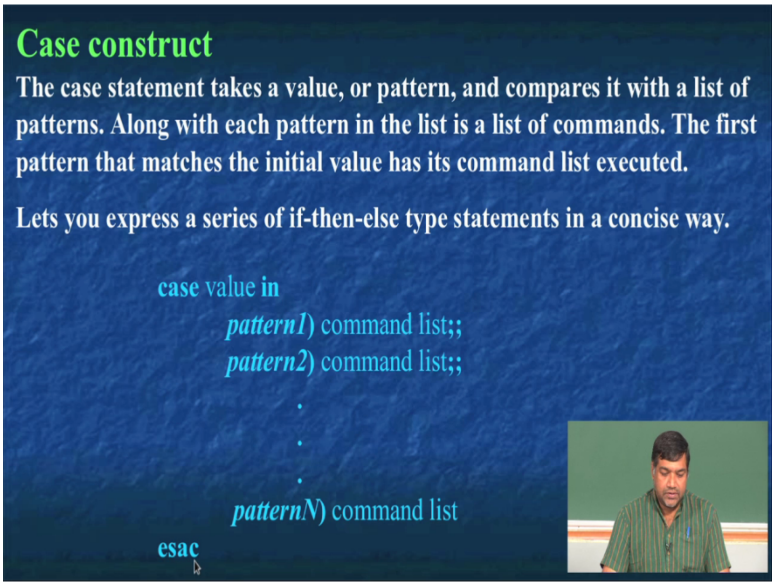
Several patterns can be separated by pipe characters,

Can match any one pattern and the correspond command list executed

Patterns are checked in order.

If you look at this case construct ok so you can either have one pattern so if you look at the previous slide we have talked about using a single pattern.

(Refer Slide Time: 03:58)



Case construct

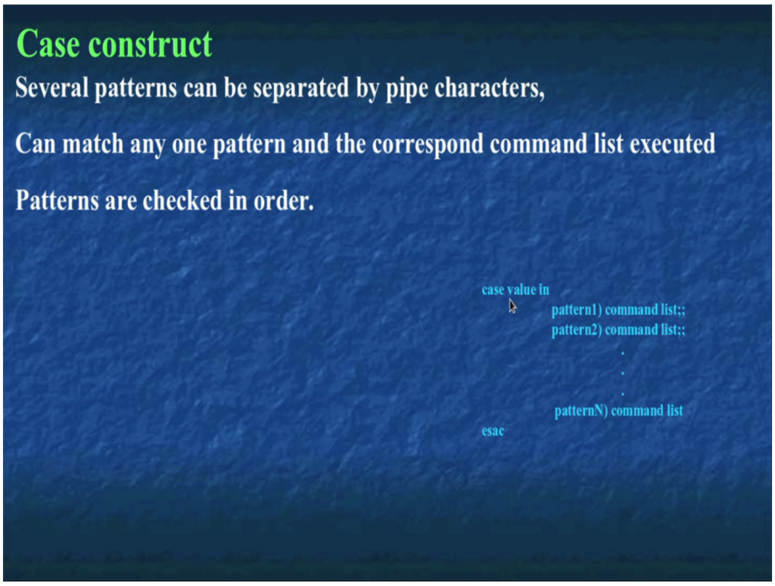
The case statement takes a value, or pattern, and compares it with a list of patterns. Along with each pattern in the list is a list of commands. The first pattern that matches the initial value has its command list executed.

Lets you express a series of if-then-else type statements in a concise way.

```
case value in
        pattern1) command list;;
        pattern2) command list;;
                .
                .
                .
        patternN) command list
esac
```

So if you look at this there were multiple patterns say we had pattern 1 pattern 2 pattern 3 and so on. And each of these we can choose by using an OR symbol. Say for example I can put a slash and say that I can have either pattern within this we can have sub patterns, let me put in that way. Let us call them as sub patterns. So within pattern 1 itself we can have bunch of sub patterns we will see all these things with an example but any how we have to slightly understand what this case statement is all about.

(Refer Slide Time: 04:30)



Case construct

Several patterns can be separated by pipe characters,

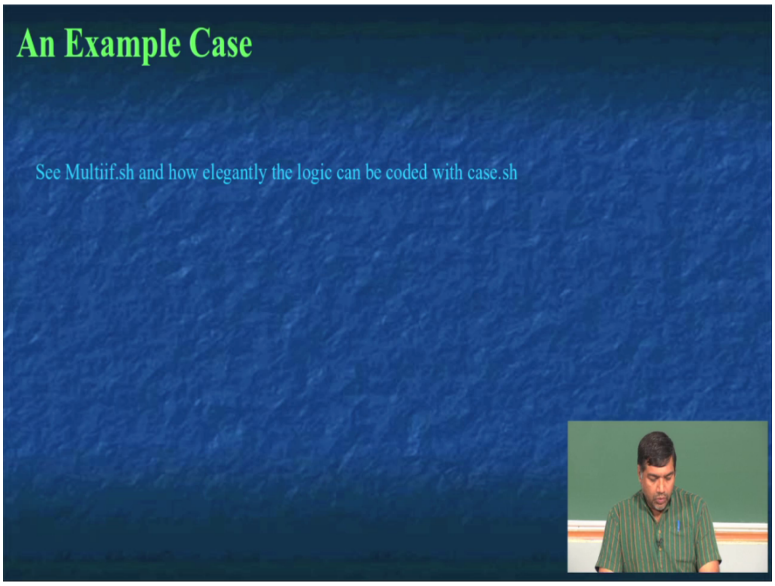
Can match any one pattern and the correspond command list executed

Patterns are checked in order.

```
case value in
        pattern1) command list;;
        pattern2) command list;;
                .
                .
        patternN) command list
esac
```

And so what we can do is that if you use a case statement ok several patterns can be separated by pipe characters ok, so if you put a pipe character here in this sense it says that you have to choose either a sub pattern 1 or sub pattern 2 and so on. And what happens with that if it matches any of the sub patterns ok under the particular command list is executed and the patterns are always checked in order, ok? So from left to right the patterns are checked so let us say for example in this case so we have pattern 1 and this pattern 1 can have sub patterns, when you have the sub patterns then you can go and check the sub patterns 1 sub pattern 2 sub patterns 3. So any of these 1 sub pattern matches then you go and execute this command list, ok? So that is a way a case statement works.

(Refer Slide Time: 05:28)

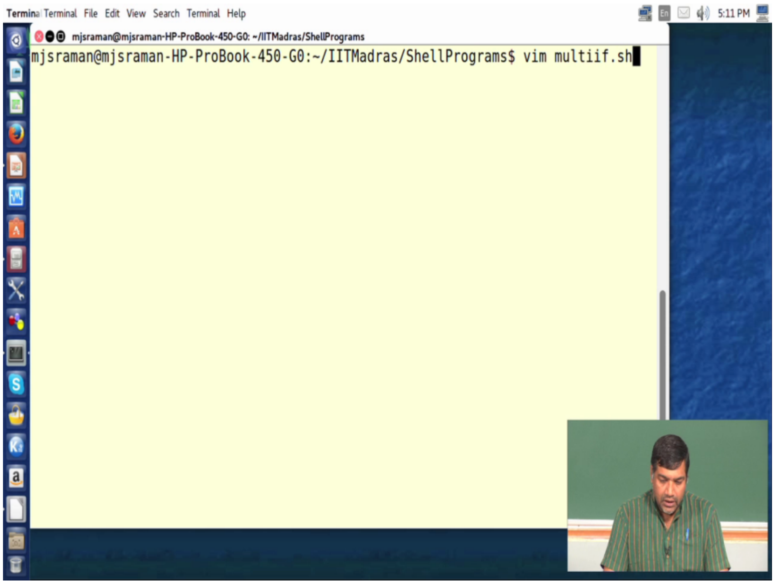


An Example Case

See Multiif.sh and how elegantly the logic can be coded with case.sh

Now what we can do is we will now take an example and then what we will do is? We will first write the example using the if then else statements see how complicated it becomes and then finally we will use the we will write the same example using a case statement. And see how it simplifies our task.

(Refer Slide Time: 05:55)

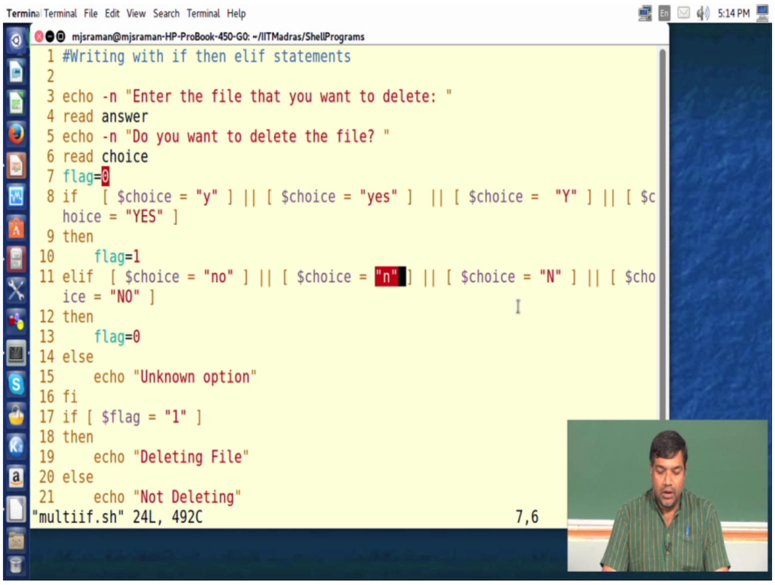


```
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim multiif.sh
```

Terminal Terminal File Edit View Search Terminal Help
5:11 PM
mjsraman@mjsraman-HP-ProBook-450-G0: ~/IITMadras/ShellPrograms

So what we will do is we will take this program called multi if dot sh.

(Refer Slide Time: 05:57)



Termina Terminal File Edit View Search Terminal Help
5:14 PM
mjsraman@mjsraman-HP-ProBook-450-G0: ~/IITMadras/ShellPrograms

```
 1 #Writing with if then elif statements
 2
 3 echo -n "Enter the file that you want to delete: "
 4 read answer
 5 echo -n "Do you want to delete the file? "
 6 read choice
 7 flag=0
 8 if   [ $choice = "y" ] || [ $choice = "yes" ]  || [ $choice =  "Y" ] || [ $c
   hoice = "YES" ]
 9 then
10     flag=1
11 elif  [ $choice = "no" ] || [ $choice = "n" ] || [ $choice = "N" ] || [ $cho
   ice = "NO" ]
12 then
13     flag=0
14 else
15     echo "Unknown option"
16 fi
17 if [ $flag = "1" ]
18 then
19     echo "Deleting File"
20 else
21     echo "Not Deleting"
"multiif.sh" 24L, 492C                                        7,6
```

I will first expand this program and then we will go ahead and look at the equivalent case program with a case statement. So as we know in line number 3 just asks the user to enter the file that he wants to delete. We can see that this programs aims at deleting a file by asking the user whether he really wants to delete a file or he just as inadvertently typed in the file name, by mistake. So the line number 4 tries to read the answer from the user this answer is nothing but the file that you want to delete. Coming to line number 5 this program tells you really ask you I mean are you sure about whether you want to really delete this file and then it reads the choice from the user.

Now in this case the user can type any choice ok before the program accepts the choice to be just the alphabet y in small or the choice could be the alphabet yes in small l or the choice could be the alphabet Y in caps or it could be choice YES in caps. Now if that happens then this programs sets at flag to 1 this flag based on this flag we will try to delete the file or not. So if I make any one of these choices that is a small y or a I type yes or a type a capital Y or capital YES.

Now you could ask a question what will happen if I type a capital Y and a small yes? Obviously this pattern matching will fail. And therefore this program will not execute the way you want it to execute. So here we are facing a problem so sometimes when we type with our hands sometimes it so happens that I could use a Y which is caps and then another e which is small and then s could becomes caps and so on.

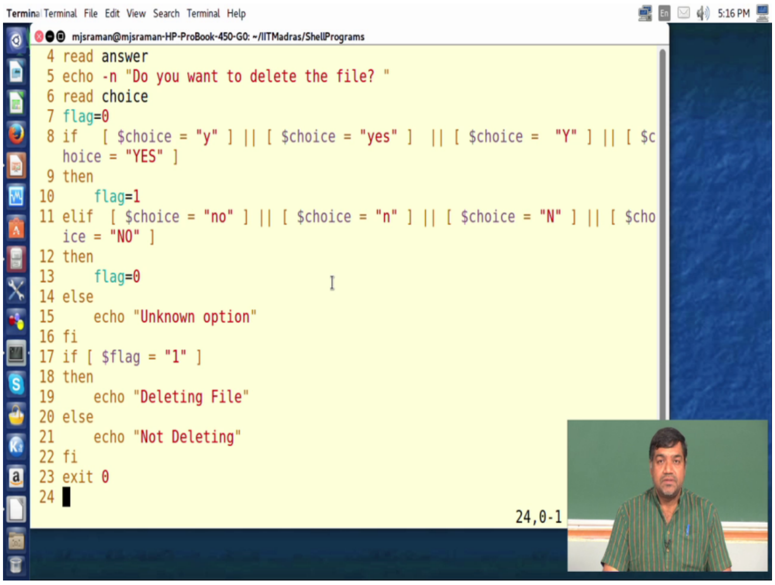
Of course all these problems can be solved by using pattern matching in the shell ok and so that is why if you look at the case statement it is actually expects a pattern whereas in the if statement the condition within the if statement you actually expect a string. So all these guys like y , yes are compared as a string and then what happens is that flag is set, now if you do not give these choices ok as y, yes capital YES then the program expects that you use something like no that is small no or small n or capital N or capital NO.

So these two conditions are usually exclusive that is if I give a y then I cannot give a n ok and if I give a N I cannot give a Y. So look at this the way this statement has been framed is so I will have to choose only 1 either I choose this path in the program or I can choose this path in the program or I will be able to by default I will be able to choose this path in the program.

So all these three paths are mutually exclusive to each other and what we do in this program is? so if the choice is yes then I set a flag to 1 and if the choice is no then I set the flag to 0. Essentially if you see I set the flag to 0 so this is somewhat redundant if I make no as a choice this is this comes in terms of program optimization we will not go into all those details but what I just want to point you is that these three paths that we see that is the first if path the second if path which is here and the third else path or mutually exclusive.

So here we have got three multiple paths three paths and out of these three paths we have to choose one path so this whole code depending on what the user types will choose one path and based on that path on flag we are setting up a flag and if you look at this flag then final we come to line number 17.

(Refer Slide Time: 10:06)


Terminal Terminal File Edit View Search Terminal Help
5:16 PM
mjsraman@mjsraman-HP-ProBook-450-G0: ~/IITMadras/ShellPrograms

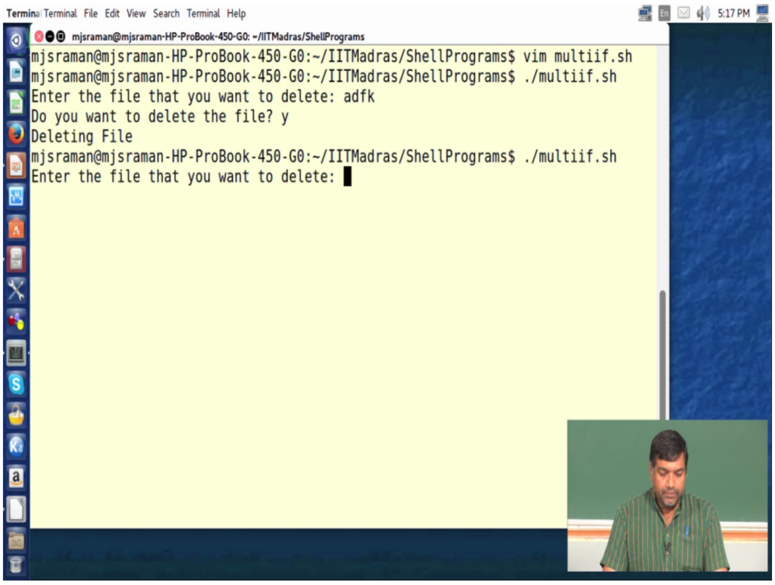
```
read answer
echo -n "Do you want to delete the file? "
read choice
flag=0
if   [ $choice = "y" ] || [ $choice = "yes" ]  || [ $choice =  "Y" ] || [ $choice = "YES" ]
then
    flag=1
elif  [ $choice = "no" ] || [ $choice = "n" ] || [ $choice = "N" ] || [ $choice = "NO" ]
then
    flag=0
else
    echo "Unknown option"
fi
if [ $flag = "1" ]
then
    echo "Deleting File"
else
    echo "Not Deleting"
fi
exit 0

                                                     24,0-1
```

Where we say that if the flag is 1 so reverting back to this logic we see that if the flag will be 1 if the choice is any one of these 4. Then here I am not deleting any file I am just saying that I am just echoing deleting a file because I do not want to I mean neither (())(10:23) this just a sample program therefore I just want to show you that how this syntax can be used so I just echo deleting a file if the choice is n or No or whatever it is from this class no I am just printing I am not deleting the file and then I am exiting with a value of 0. So let us now try to execute this program and see how it behaves so we can at least have three minimal test cases.

The first test case can be we can type either y or yes or whatever it is. The second test case is we can have n or no or whatever it is and the third test case we can type any of the values like capital Y, small e and then capital s and so on or any other input and let us see how this program behaves.

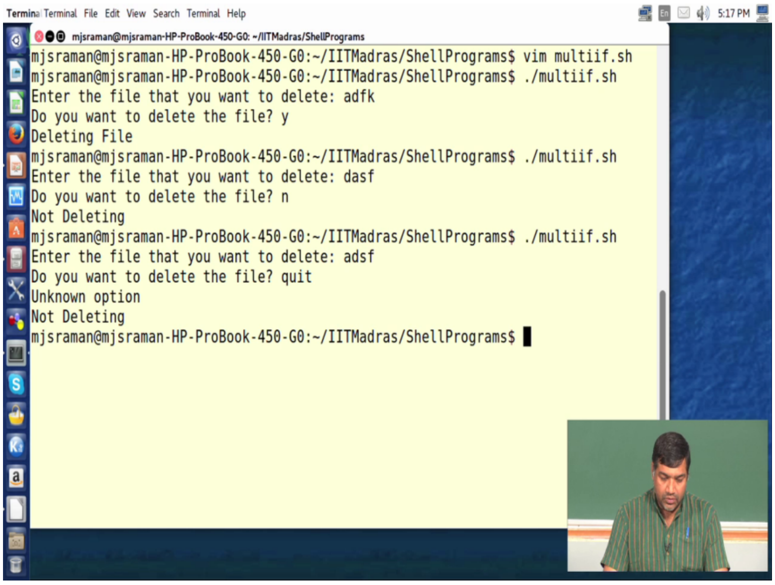
(Refer Slide Time: 11:06)



```
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim multiif.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: adfk
Do you want to delete the file? y
Deleting File
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: 
```

So I will run this program so let us first in the first case I give some file name and then I say I want to delete the files. So I am going to give a small y, so this is the possible test cases and it says deleting the files so the path followed is correct.

(Refer Slide Time: 11:20)

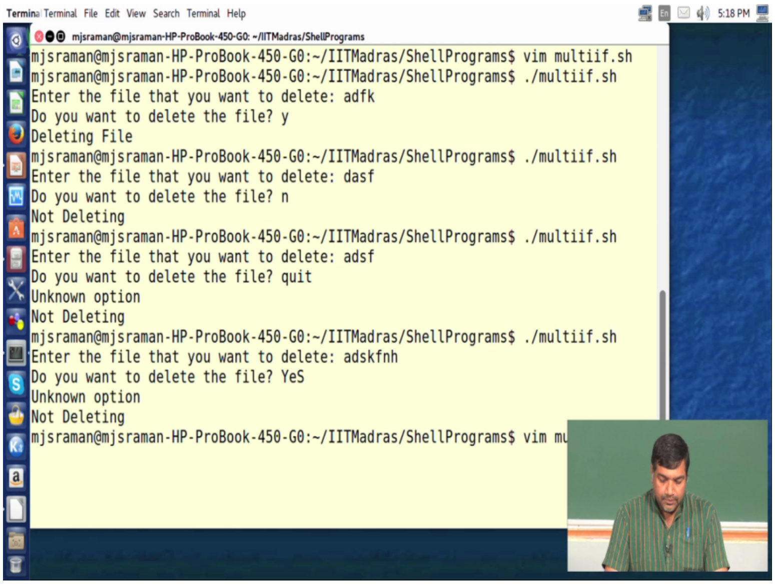


```
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim multiif.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: adfk
Do you want to delete the file? y
Deleting File
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: dasf
Do you want to delete the file? n
Not Deleting
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: adsf
Do you want to delete the file? quit
Unknown option
Not Deleting
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ 
```

Again I run this program ok and I give some program to delete some file to delete then I say do you want to delete the file I put n so it says it is not deleting the file so this is according to whatever logic we have followed. Now I am going to try the third test case where I am going to type the filename and I am not going to give either y, n or some of those choices I will say

I choice like quit ok so it says unknown option and because it is an unknown option it says now deleting and it comes out.

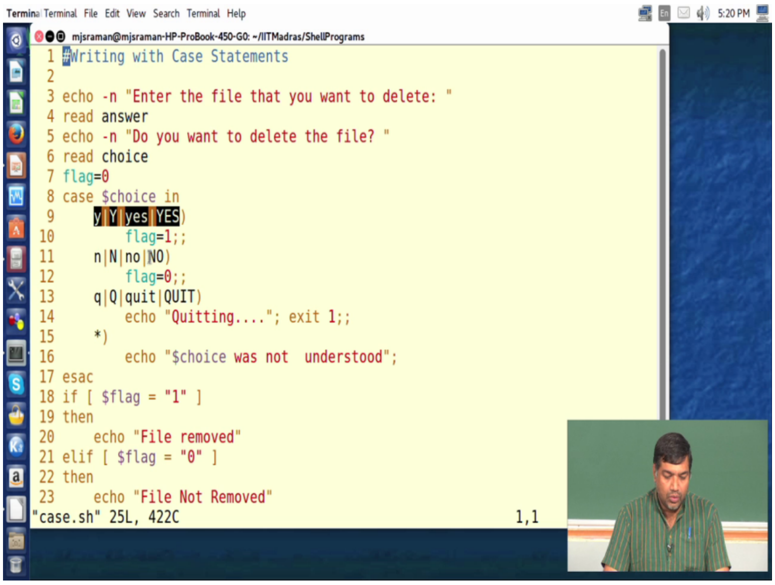
(Refer Slide Time: 11:53)


Terminal Terminal File Edit View Search Terminal Help
5:18 PM
mjsraman@mjsraman-HP-ProBook-450-G0: ~/IITMadras/ShellPrograms

```
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim multiif.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: adfk
Do you want to delete the file? y
Deleting File
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: dasf
Do you want to delete the file? n
Not Deleting
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: adsf
Do you want to delete the file? quit
Unknown option
Not Deleting
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./multiif.sh
Enter the file that you want to delete: adskfnh
Do you want to delete the file? YeS
Unknown option
Not Deleting
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim mu
```

Now comes the next test case where I give yes sorry I will give the file name and then I give yes now if you look at this it says unknown option and deleting the file now this is one of the drawbacks if you try to hard code anything into your code. See essentially you should keep any coding as generalized as possible. So here in this at least in the shell what we have to do is if you look at this line ok?

(Refer Slide Time: 12:24)


Terminal Terminal File Edit View Search Terminal Help
5:20 PM
mjsraman@mjsraman-HP-ProBook-450-G0: ~/IITMadras/ShellPrograms

```
#Writing with Case Statements

echo -n "Enter the file that you want to delete: "
read answer
echo -n "Do you want to delete the file? "
read choice
flag=0
case $choice in
    y|Y|yes|YES)
        flag=1;;
    n|N|no|NO)
        flag=0;;
    q|Q|quit|QUIT)
        echo "Quitting...."; exit 1;;
    *)
        echo "$choice was not  understood";
esac
if [ $flag = "1" ]
then
    echo "File removed"
elif [ $flag = "0" ]
then
    echo "File Not Removed"
"case.sh" 25L, 422C                                       1,1
```

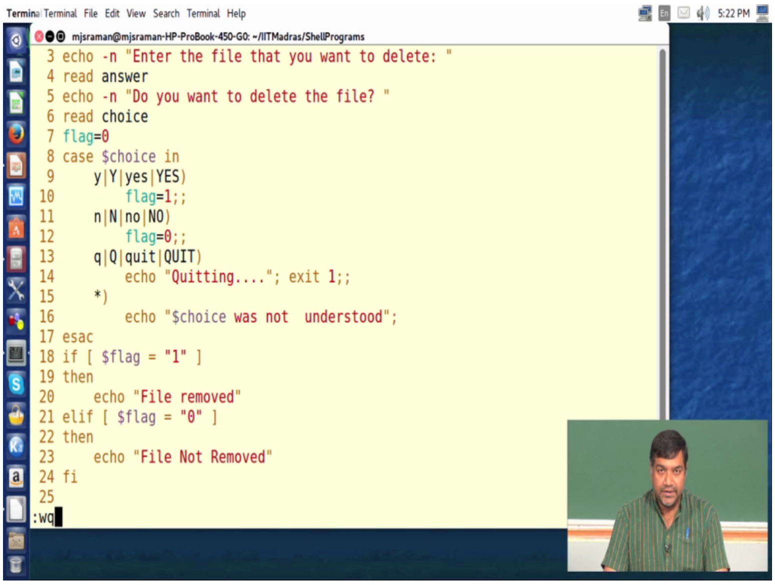
So we are actually specifying a concrete alphabets or a string, ok? And this is actually sort of a bad practice you should actually put here specifically a pattern, ok? which can accept any of these variations of yes or y and you should also say there is a pattern now we will leave that as an exercise to you, ok! So what you should do is when you are using the case statement instead of having concrete strings like yes and all that you should try to put in a pattern ok? So that the code can do a pattern matching and then take a decision.

I mean this is slightly difficult but if you can understand the use of question mark or plus or dot or star etc, you can use one of your terminologies use the any one of these methods and try to get clue that we are giving you but you can let us come back to this so what we will now do is we will now this program actually looks I mean actually syntactically it does not look nice let me put it that way I mean it does not look elegant. So what we will try to do is we will try to write the same program using the case statement so we know what is the syntax of the case statement is? So let me now take the case that sh file. So in this file you look at this the first five lines are the same there is no change but then look at line number 8, ok?

So in this line number so what we are doing is we are reading a choice from the user ok and then this choice can either by Y capital, yes or YES or this can n, capital N, no and I also added another choice for quitting ok? And finally I am putting a pattern matching here. This tells you that if you type anything other than what is given here you see if you type anything I mean this star has various users in shell script at one point it is used for multiplication symbol on epoint is used for character expansion and so on. So in this case for example if you type any characters or strings other than what is given see either y n or q then it will say that the choice was not understood.

Now remember even there we provide some provision but look at how small the code is ok first thing the code is actually it comes about 7or 8 lines I mean 10 lines approximately and it looks more elegant so let us see what this code does. So if I make a choice that is whatever I type if you remember first we type the file name second we are giving a choice so if whatever choice we give then if this choice belong to anyone of this then this flag is set ok you put double semi colon the reason you put double semi colon is in shell you can terminate a statement with a single semi colon. So in order to differentiate that termination with the termination of a case a choice of a case statement in a path we are actually putting a double semi colon, of that is the syntax that is you have to follow again if I choose y, Y, yes, YES then I set the flag to 1 if I choose any one of this I set the flag to 0 and then if I type q it says quitting and if I type any one other than this whatever I given it is supposed to say choice was not understood. And as usual the logic of the program has not changed if you remember the previous logic we started with dollar flag. And then we looked at how to edit or remove the file or not.

(Refer Slide Time: 16:10)



Termina Terminal File Edit View Search Terminal Help
5:22 PM
mjsraman@mjsraman-HP-ProBook-450-G0: ~/IITMadras/ShellPrograms

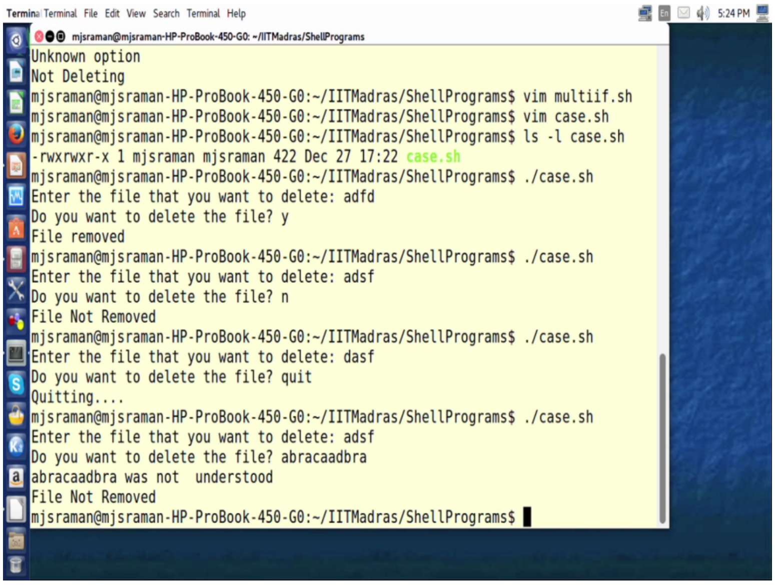
```
echo -n "Enter the file that you want to delete: "
read answer
echo -n "Do you want to delete the file? "
read choice
flag=0
case $choice in
    y|Y|yes|YES)
        flag=1;;
    n|N|no|NO)
        flag=0;;
    q|Q|quit|QUIT)
        echo "Quitting...."; exit 1;;
    *)
        echo "$choice was not  understood";
esac
if [ $flag = "1" ]
then
    echo "File removed"
elif [ $flag = "0" ]
then
    echo "File Not Removed"
fi

:wq
```

Here we have use the if else statements ok just to show and one way to execute it is if you can just put even if you can put a else statement which I just tell you that because this flag is 0 if the file is not removed ok?

Now anyway we will run this program so let us run this program and we will run all the three test cases in this case we will have 4 test cases because we have added a new choice of q, ok.

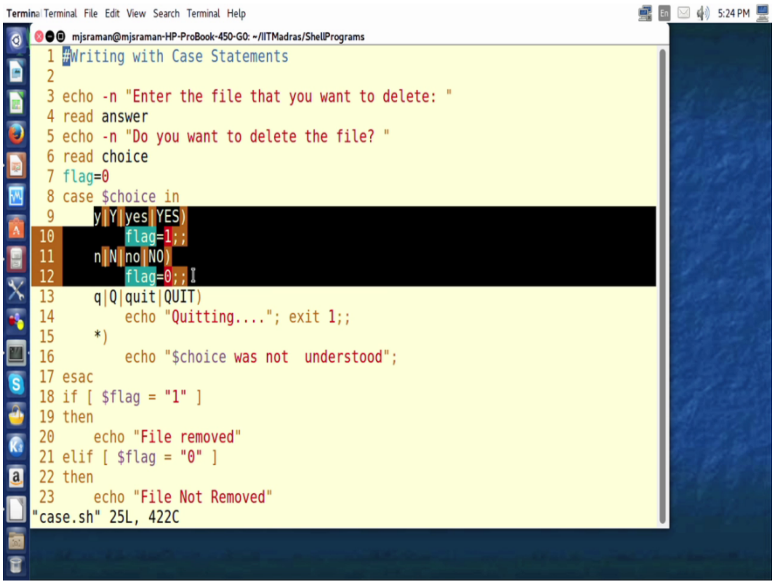
(Refer Slide Time: 16:38)



```
Unknown option
Not Deleting
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim multiif.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim case.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ls -l case.sh
-rwxrwxr-x 1 mjsraman mjsraman 422 Dec 27 17:22 case.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./case.sh
Enter the file that you want to delete: adfd
Do you want to delete the file? y
File removed
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./case.sh
Enter the file that you want to delete: adsf
Do you want to delete the file? n
File Not Removed
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./case.sh
Enter the file that you want to delete: dasf
Do you want to delete the file? quit
Quitting....
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./case.sh
Enter the file that you want to delete: adsf
Do you want to delete the file? abracaadbra
abracaadbra was not  understood
File Not Removed
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ 
```

So let us run this program, and remember before I run this program I have actually changed the mode of the file so please remember to change the mode of the file to executable mode otherwise I mean we have already crossed about 10 sessions so we tend to forget what was done in the first and second session. So because of that I am saying that you need to ensure that you have execute permissions for this file and in this file suppose I execute this statement, ok. So it says enter the file you want to delete. Ok I will type some junk file and then we remember the choices so I say y so it says file removed.

(Refer Slide Time: 17:46)



Terminal Terminal File Edit View Search Terminal Help
5:24 PM
mjsraman@mjsraman-HP-ProBook-450-G0: ~/IITMadras/ShellPrograms

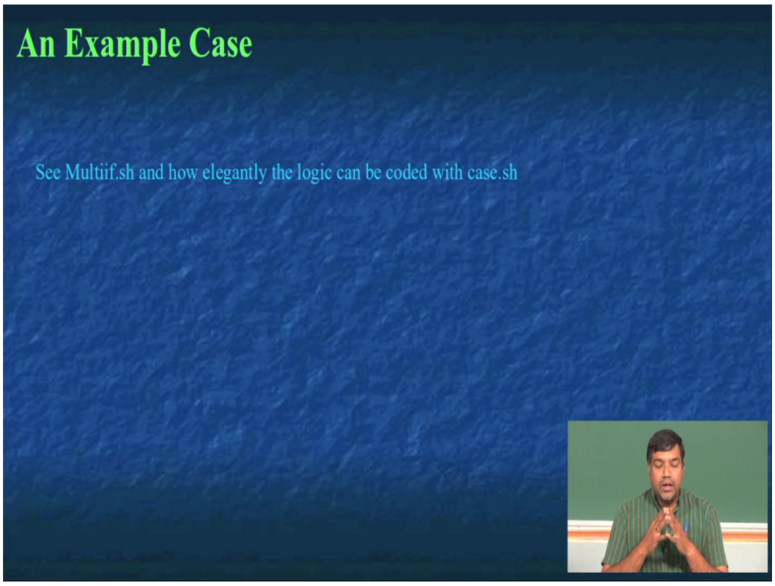
```
#Writing with Case Statements

echo -n "Enter the file that you want to delete: "
read answer
echo -n "Do you want to delete the file? "
read choice
flag=0
case $choice in
    y|Y|yes|YES)
        flag=1;;
    n|N|no|NO)
        flag=0;;
    q|Q|quit|QUIT)
        echo "Quitting...."; exit 1;;
    *)
        echo "$choice was not  understood";
esac
if [ $flag = "1" ]
then
    echo "File removed"
elif [ $flag = "0" ]
then
    echo "File Not Removed"
"case.sh" 25L, 422C
```

Now let us run the test case again I will give some file name then I give 10 it says file not removed then I will run again for the third test case which is quit so it says quitting then I will run the fourth test case where instead of quit or something I just type something abracadabra so it says abracadabra was not understood. ok file not removed. Now if you look at this there was no place in the code that we mention this word so for example if you look at this if I typed that whatever I had typed and since abracadabra was not in any of these so I actually came here this is what we call it as default.

If you know about C programming language in any case statement you are suppose to have a default and this is the default that means if it does not match any one of these paths then it takes this path. So this star tells you that you can match any string ok Now one the exercises that I was talking about was that you should use this symbols like star or question mark and then convert this such that it is not just these 4 to I should be able to choose this path even if I type capital Y small e and capital S. So that is the exercise that you could try out on your own and see how to ensure that the right programs which are very general rather than giving them concrete strings like this. That is slightly difficult but you can try it out so that gives you a very good exercise on pattern matching.

(Refer Slide Time: 19:07)



An Example Case

See Multiif.sh and how elegantly the logic can be coded with case.sh

So remember one of the things that i want to insist here is that we are talking about the tools ok how we make use of that tools in your examples or the logic or when you make a sequence of statements ok and then make them work together is your ability and that is an art so coding so all these tools can be learned but then how to use these tools to solve a particular problem is an art and that is the reason in between we are discussing lot of exercises also. So from the next session we will look at other features of shell.

Thank You!