SEER AKADEMI
LINUX PROGRAMMING
AND SCRIPTING-
PERL

After like two minutes and then just check the audio with the audio hi everyone again welcome to this course on the Linux programming and scripting we covered the first two modules the basics of Linux and then the Linux reporting last time we covered the second part of the of the file system that we were study we studied two file systems the Sun NSS as well as Andrew file system.

The Andrew file system is what we powered the last lecture we also went through all the differences between the Andrew file system and the denoted certain similarity certain difference similarity was that you so some kind of a virtual file system manager or the Andrew file system basically goes more like a process running which handles all this virtualization so for the NSS we have to mount those PI systems down separately.

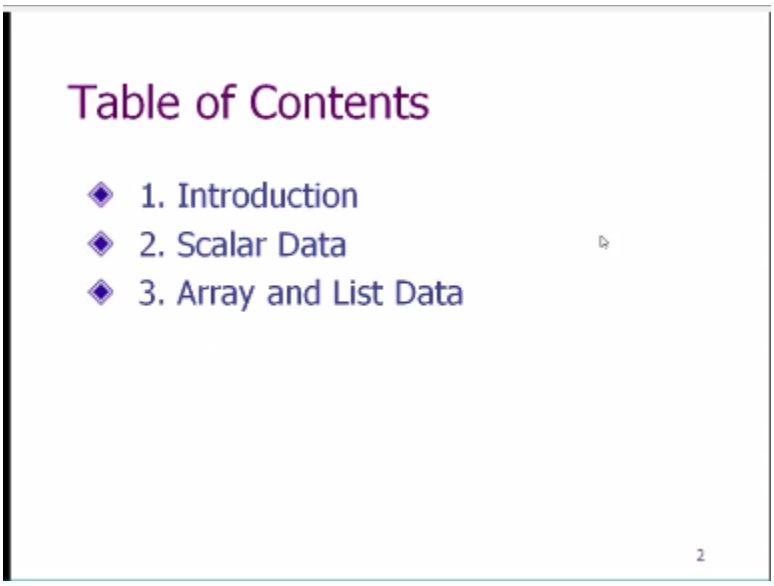
And then the control was more at the client side for the NFS system whereas in the server side or we handle file system so the Andrew file system is more of a time bomb more secure and we also saw that the Andrew file system we do not need to actually there are not like many Andrew files set up basically they came in to the subnet or sub Network kind of systems.

Systems are not there whereas the NSS is more distributed as some clusters so I mean there are like few Andrew PI system some services available today and the others are actually here and the PI system basics course a mobile standardization that happened in distributed file system media I just want to keep one of want you to keep that in mind so today we are going to switch gears and talk about the programming aspect of the course.

So we will be starting with the simple language called pearl and this pearl language is used widely in the industry especially in the high-tech companies in fact lot of software companies also use this particular language the Perl language the hardware companies or all kinds of scripting the useful language we will learn about why this is used widely and what makes it more attractive than the traditional programming languages or tram or c or c plus.

Again Perl has many variants but the most significant change in Perl is recent one because we want object-oriented code which we will touch upon towards the end of the of this material.

(Refer Slide Time: 03:47)



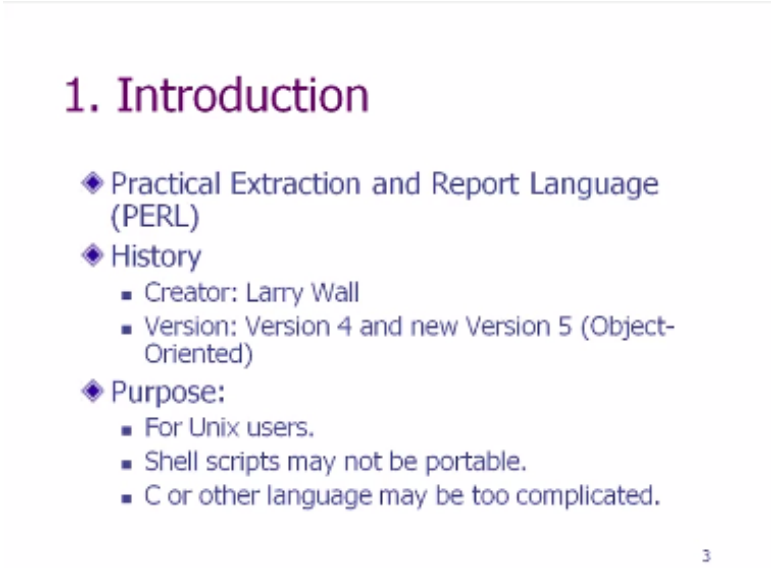
So let us look at what we have the agenda for fitting so we will start with the direction today we will be introducing move into the data side of things pearl even though we will touch upon couple of commands people either understanding some basic commands cannot understand data screen fullest extent so we will go through some of that stuff and mostly today we talk about the scalar data which is essentially a single variable or the data that holds a single value.

And then we will have also we will talk about array and this data arrays you can import that as a data structure which has form some kind of array the scalars or the scalars themselves are there are many scalar beta which is irate together so we have some commonality negative open deck things like that and then we will also look at some of the list table it is again they are also kind of erased but they are specialized arrays.

Event about that way so that is the pretty much what we will do and as usual you know we will talk about outright some basic scripting today and then as we go through the course we will progressively increase the level of difficulty and hopefully by end of this module you will all be like well change programming simple and you can write any scripts anytime with any whatever the specification that you do not even see.

Again I just wanted to say that the basic algorithms when you are writing programs they are all similar and for we have already covered those before in your own undergrad and other courses so I would not be talking about the algorithms as to how to do it you have questions we can always talk about talk it over but mostly language we will be focusing on the syntax and semantics in the Perl language itself.

(Refer Slide Time: 06:13)



So let us look at the full list an introduction to go pearl stands for practical extraction in the code language so if it is a big mouthful lecture people always refer to this as well and if for people like I mean who like the language it is the they also talk all it has a pathologically eclectic rubbish Lister and then but I was like I mean some people who still do not like pearl they call it like pointless exercise in redundancy and just in this is only submitted they think that that is redundant.

So anyway depends on which group you are if you are well versed in other languages like C C++ and you are always like so hands-on with those languages you would not find this that useful probably will still go to it both languages but if you are more like me who is essentially like I mean I programming or following some of my problems then you will like Perl better because it gives you a very easy and very quick platform to do you program.

On the history front the creator of pearl is Larry wall so and then the versions got updated the perfect is a new enhancement and version or is pretty much the standard version both people use and then there is also a new version5 which is more object oriented so if you look at your text books essentially like I mean it will tell you that there it is a based on the version4 or version5.

And I would recommend you to actually at least look at the version 5 books because that has the object or construct for all the potion and then the main purpose of pull why he developed that is for the unity rules so in UNIX you get like two kinds of program that be softly one is this shell scripts which touched upon it briefly as to how to do then I am just whipping with the various commands from the arc grab all those things basically that we can use it shell scripts.

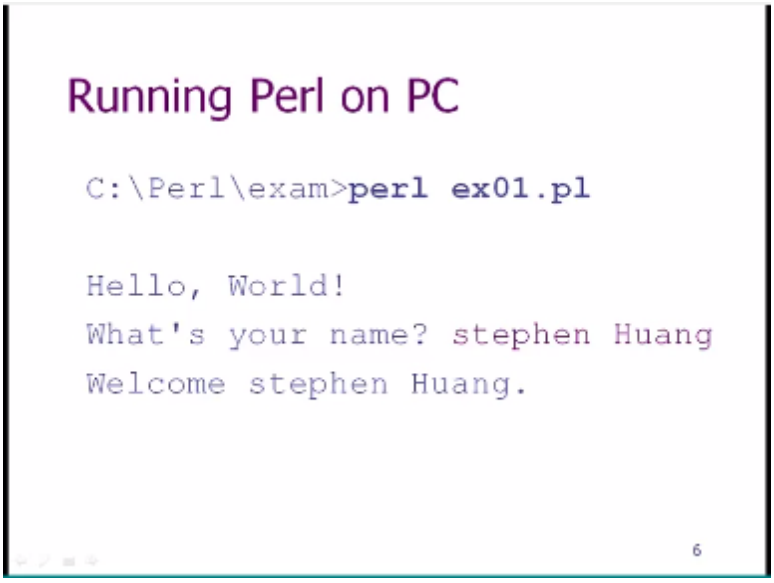
The shell scripts are cumbersome to write and they are not really portable so that is one reason they are looking for a formal language the formal languages like C no more complicated.

Strongly typed languages it is not really useful for people in other areas like say if I am going to be elevate design to do some program so in VLSI design as you know want to augment your work with programming rather than the programming itself becomes part of the new world.
So in that kind of scenario you need a lightweight language jazz some formalism not like a script but at the same time you do not want like too much formalism like any other language so that is the reason why people want object or pearl.
(Refer Slide Time: 09:58)

So here is a simple script essentially like I am I just wanted to highlight a few things here ,so we know these things basically would not move #! bang as bangs essentially to tell the shell that water is following that in the file script then we have this user bin pearl which satisfies the path to the pearl suitable which now like uses this as a main interpreter and then it starts in Perl these are all linked and consecution in Perl.
So this itself is a command and then followed by argument and here notice that the code here thing you can go inside the code and then there is a \ N and then at the end there is a ; so I just wanted to just observe these things at this point it will come back to it in a while and why they are and what is the significance of each other again in this stage this one again is another print statement say what is your name port and the ; and then we have another special stuff.
It is $ sign and then name then again here also there is another special < and >and then there is something in between which is STDIN and then here there is another command chop command. For this one there is no; but instead this is for a parent and then + the $ game the same thing that, okay and then finally another print statement with some more arguments so if you look at this particular statement this may be it may be familiar with this construct which is very similar to

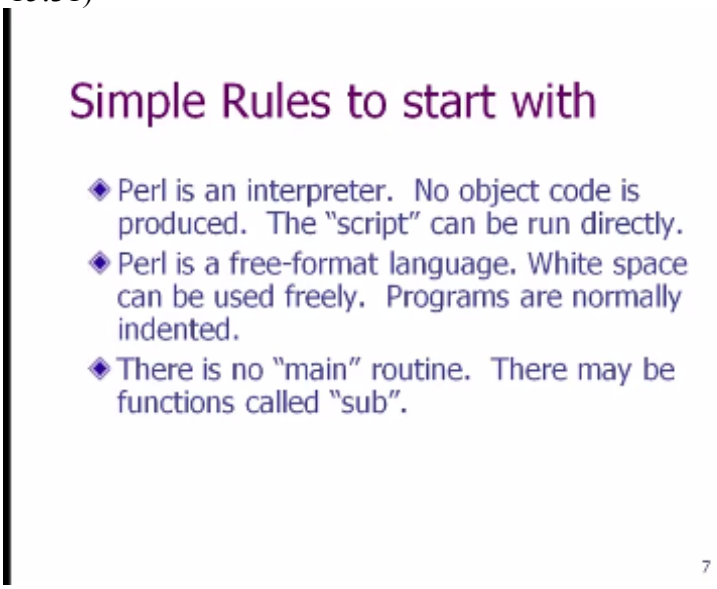
like a steel fence for where you are calling a subroutine with a subroutine name and an argument and if you just leave this as blank then you walk into that one.
So anyway we will come to this in a minute but so this is a simple program what it does is this we will see in the next slide or what exactly to do so here we can execute this program in the UNIX system again you know some other things basically.
(Refer Slide Time: 13:21)

Again here you know that actually the percentage is the shell it stands for the shell and then you type in the hello doc pearl which if you put the map and before you give this you need to make sure what this is the execute permissions are set that is the change more than 777 more stain mode at least some one at least you can have one or + X it is filing so that you can execute it directly and here the way it works it I s essentially it prints out.
This hello world this is the first statement that we saw and then it goes to the next line and then asks for or to your name and then it waits for the user to input this portion so the user types in is named immediately is a welcome the usual so we saw there actually like this particular input is captured under $ name you and then the $ name is 1 what entered these are the things that we saw in the earlier time.
(Refer Slide Time: 14:51)



So let us see more about all itself so if you are running it on the PC actually then you have this keyword pearl and then you can actually run the full with this one instead yeah it is called ex-01 but it would be also hello pearl in the same thing it also does the same so this C term knowledge within terminals like Windows and then there are a lot of versions available to install Perl version available and then one.

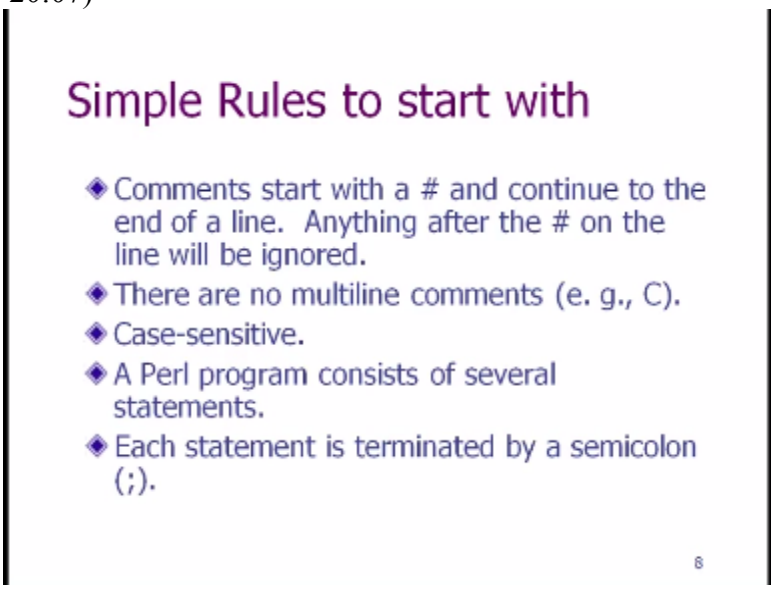
(Refer Slide Time: 15:31)



So now let us talk about some simple rules that in purl that we will begin with so number one is pearl is more like an interpreter so in a compiled language like C or C++ you have to first compile the program into an executable and then the executable is can be run that executable. Is cannot be read by a human and I read by humans and it is more like a machine or and there the compiler does lot of optimization to come up with the best score possible.

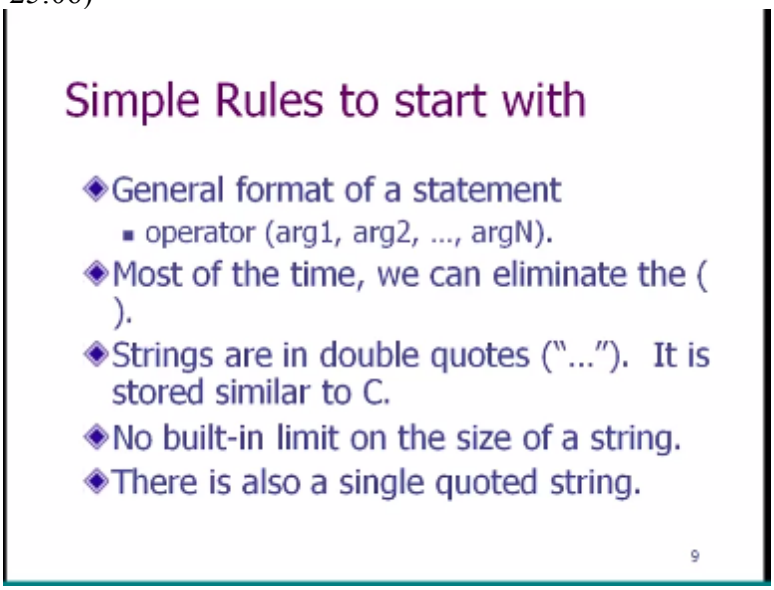
Which also has minimize the runtime and thing for so the compiler courses a separate course and essentially urge you to take it when you get a chance because that will teach a lot of techniques that the modern compilers used to eliminate redundancy and eliminate the loops and things like that but or Perl actually you can think of this as an interpreter it interprets line by line and then basically a group of such lines directly.
It does not need to generate anything in intermediate formats and this is a great help because this is the one of the reasons why engineers like this especially electrical like pearl because it is more like what you see is what you get the speed it and just over the program and just run it that is it this way you do not need to compile with it you are going to compile it you do not need to optimize it method whatever you write immediately executes.
So that is one good thing and then goal is also a free format language so essentially the format is not dictated by where the white spaces or how the white spaces form so you can actually use white spaces or feelings it is one white space on anyway it does not care the just as you have seen like the earlier example the commands are terminated by; that is the reason why though put semicolon before command.
And essentially like the indentation of programs we do it only for human readability purposes it does nothing to do with better Perl needs some indentation or something you will see that there is some difference when we talk about language like Python which is not as free format and then this is where the whitespace means certain things even in tickle actually the whitespace another thing.
So we will see like that put white space where multiple web say in both those languages for the coming sections coming down week but right now I want you to think about pearl as a free format language and you can use white space form totally free and then another thing is Perl just starts executing from the top this one by line the line it just goes there is no concept of a name so whereas like coming in language like C.
Actually always so only the main routine will get executed there everything if it is in the last section it just executes the name from the classic and then essentially inside main you can put a lot of functions subroutines and then basically we want that subroutine and actually running it as subroutine still suppose it in fact some of these the Perl libraries themselves comprises of a lot of subroutines or we need some of for example.
With the Chop command that we saw it I s a classic subroutine type of a command essentially which may be like then split it into multiple different other commands and then that may call

some of the commands which essentially serves as top function we will look at what that function does certain so just wanted to.
(Refer Slide Time: 20:07)



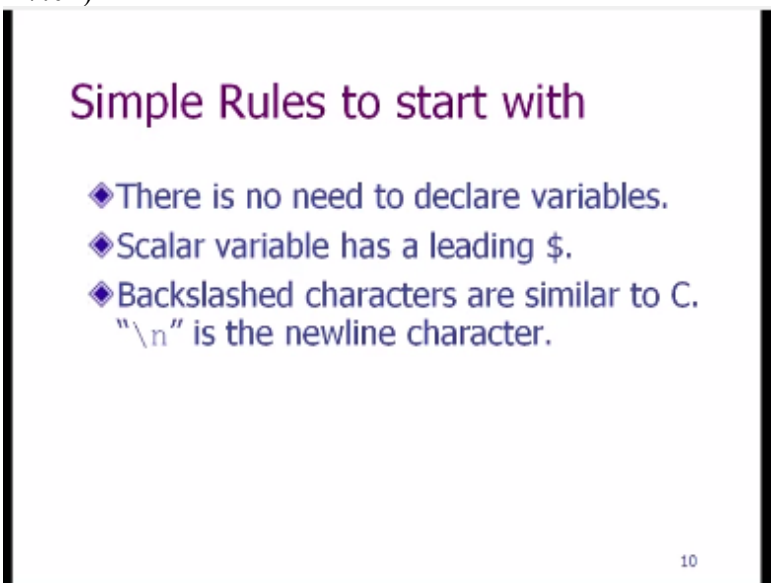
Let you know #! this one then being the more to incorporate comment actually the command starts with #tag and then they continue until the end of the line so every line which starts or at some point there Is a # is treated as a command line only exception to this rule is the first line which we saw there actually it started with the #! The it is together with one command which tells the program that he this is the it identifies this particular file as a Perl program.
So that is only reason why we use that and that had needs to be the first long time so that is all me no restriction there so anything after that # until the end of the line will be so you can actually put a # at the end of a particular command once you close the command with the # you can put a # and bite what with the vendor and it is perfectly legally only thing is like you need to make sure that the command is closed there are some commands.
Actually like and I will take it back in any cases you can start with the # and discontinue the only restriction is when you are missing many languages together like the program with Perl and things like that there you need to follow some additional rules but in the pure Perl language any time you start inside the line with the # whatever is the remaining of the end of the line.
So in C actually we have these multi-line comments and I think you must become agree with that basically they take a look in C we can do this form like put a / and then * and then blah. And then put another * and then a / this entire thing is a comment in Perl actually it is not like that basically we need to# and then whatever is that line would not abandon the other have oh it cannot be this come back.

So this one you and then the other rule in pearl language is that the names are all case sensitive so the uppercase name and lowercase name sensitive so again that if you are writing a program make sure that you stick with one case and essentially like you have to that is the only way that and a Perl program itself instances of several statements so we already saw like at least three types of statements.
The one was the print statement the print actually had an argument with it folks will revise that and then we also saw the Chop command which is basically your document was within a within parentheses then the third type is that we thought was an assignment where we say like the $ name = < a CDI in > a pound that is another type of statement.
So we have like several of them visiting more number of statements as we go through this lecture and then every statement is terminated by a semicolon so that is the reason why also like I mean this is the language itself sis form a format free because the you can terminate the end of the statement is only becoming by to form so the only exception to this rule is when you have comments so just keep that in mind or when you have the comment Which is like starting with the # you do not need to terminate it with the ; because it is technically not a statement.
(Refer Slide Time: 25:06)



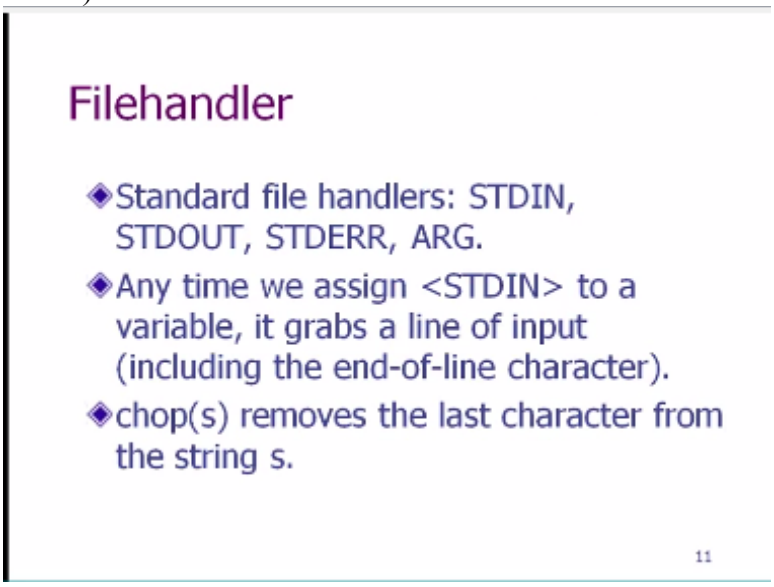
Some more rules the general format of any statement will be an operator followed by a number of arguments and usually the arguments are enclosed within the parentheses.
So but some sometimes we can actually are even more for time you can eliminate that potential so you do not have to actually specify the path and anything that is within " " is a string and it is stored similar to C with its stored as a string.
The string is also an array of character sequencing and then pearl itself does not impose any built-in limit on the size of the string but sometimes you are the regular operating system

painting for something so you have to when that limit is reached actually so the thing is basically you can read the entire book with all the characters inside the book as a single string so you can think of your pearl book which probably has like more than 500 pages and from.
The beginning of the page the end of the page the entire string you can do when you entire character set you can think of that as just a single string inside both provided the operating system allows so pearl does not put any restriction with the operating system and there is also a single coated strings even see the distinction between the code string Thing we have got exchange and exchange with ""in a later stage I want to tell you the all things I am at this point.
(Refer Slide Time: 27:01)



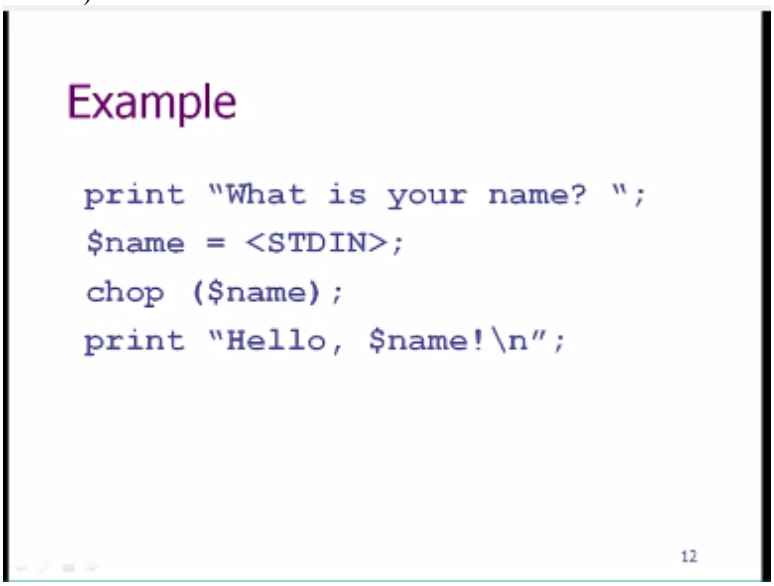
And then the biggest advantage of all is there is no need to declaring variables and this is one of the reason for big reasons why people love pearl in other languages so if you do not declare and start using available the program will start complaining here you have not located the data we you cannot use it used by somebody else so it can actually do some move with a particular playable live and things like that some like that.
Analysis here in Perl you can just use it as you go and then basically the tool will figure out how things feel one portion I will say so and still recommend you to declare the variables.
At Least set it up what variables that you are using in within a program at the beginning of the program the reason is it is just painful to use need to declare because you may use some form of work on certain things.
And the person who is actually reading it may not understand what kind of variables he used there so it would be a good idea good practice to actually declare it open and then use them basically so again it does not impose so maybe like the things like the loops and stuff like that

you may not need to declare those variables like for I equal to 110 I even with this fine box but in any case any other situations.
Like going an algorithm and you have a way to solve the problem that those variables if you specify the form that really helps the next person who tries to understand how you wrote before and then the other thing that we saw of this $ sign in the first program so any scalar variable is started with that terms.
So if you write a $ a $x any other kind of $ something all knows that that is available so and it is also it is also a scalar variable means like that variable can hold only one value and then the \ characters there they are similar to free programming so for example the \ n to the newline character \ T is the tab so you can actually create it like lot of this and that is a \ even special characters you want to own make a make sure that they can print You need to make sure that that is escaped with the \ so we just call it make escape character but it is the \ .
(Refer Slide Time: 30:06)



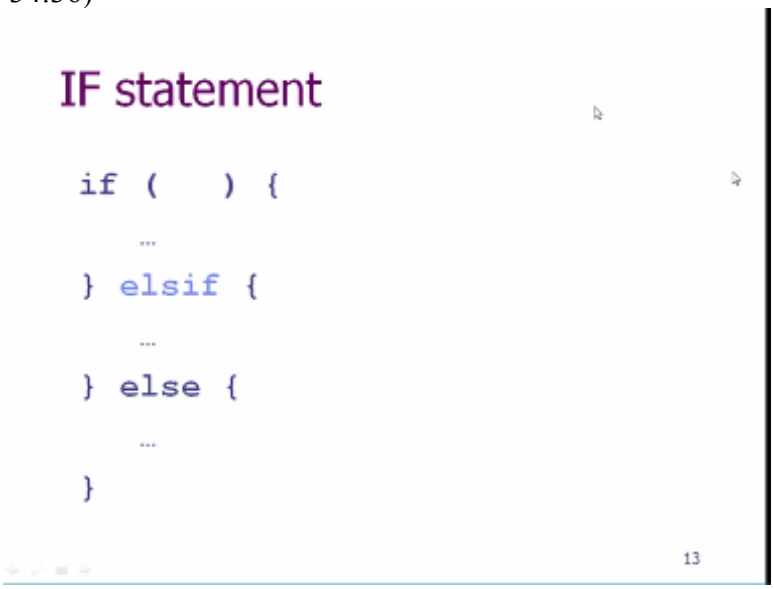
Then there is a concept of file handlers and they are open standard file which comes by default with pearl installation the standard file handlers are a STDIN and STDOUT, STDERR and ARG. The a study in stands for standard in which means that the file that is basically there whatever you are typing in the keyboard gets logged essentially it is not logged actually that file that operating system uses.
That files address is the standard and then the operating system wherever it puts out this particular file or as an output that is the standard out and then there is also like an alternate author that operating system uses to produce some output reports basically that is the a STDERR

our standard error has been also going one and ARG is a generalized file handler for getting some data so you can actually put like the and this is basically.
We feel I can go are the values that we can input and figure so how to a variable and grabs so here anytime we assign the standard in to available it grabs the line of input so the entire line is taken from the input and then assigned to this STD.
So if you think about it basically including the new line the next line going to the next line everything is captured and dump it into this STDIN so we made a scalar variable called name made that equal to this form standard in so whatever its standard in is reading that is what goes into goals still available again you may be confused because of spring itself as I told you it is just an array of characters.
So how can you are assign a spring which is more like an array to the variable or essentially just a scalar given since only one value the way we do it is essentially always treat them as string and we do not let them wander off and go out and basically or separate them into like characters and themselves and then in the case of this that is not what we are doing we always treat that as a single singular entity.
So that is why you can still use a scalar rating to focus on wall the scalar variable is now pointing to a single entity and finally be the last command chop here read the specify is S so S is the argument or chop command and here that S because like the chop commanded the last character and on from the string from the string S so but this is a useful command that are there is one more way to act in your movie for the last character We will learn about that the user lecture.
(Refer Slide Time: 33:29)



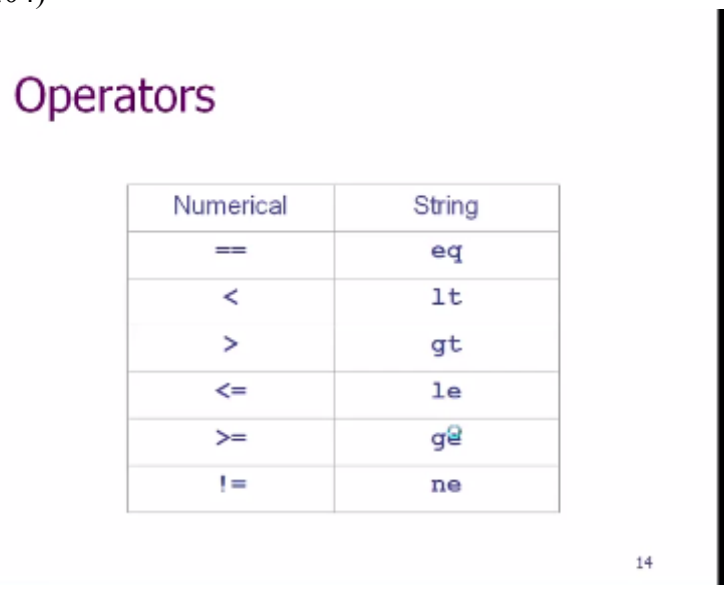
So this is the example that we saw so now like I mean much more clear there is the commands so if you look at it there are three types 123 because the three types of form statements the first

statement is a print statement basically just an output statement to output something the second statement is more like an input statement basically it gets paid for STDIN in to be input and then they see the assign name.
The third one is more like an operational statement it is within the program itself which is for it chops the last camp empower me which is typically the new lengthen and then finally the print hello is the output one which as it goes basically it prints the $ name it is this form variable that holds whatever you type in so they did it I think this is a good little exercise that will help you hone in some of the skills.
(Refer Slide Time: 34:36)

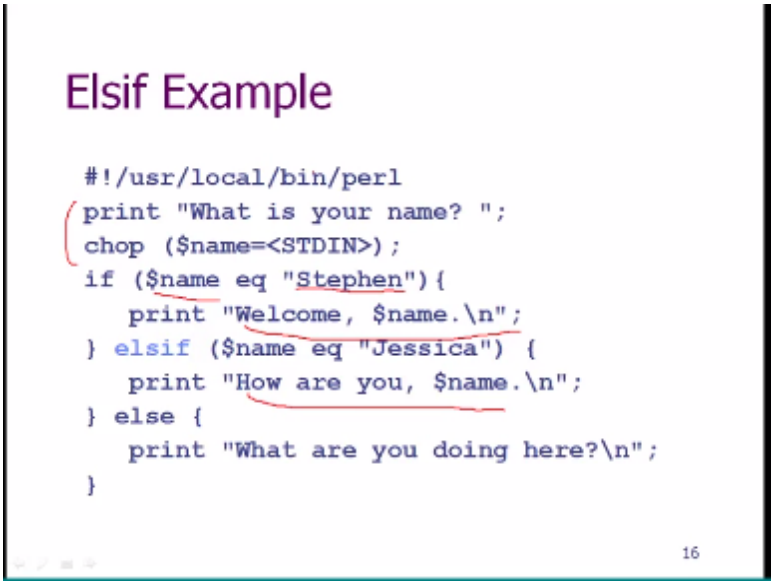


So before we go into the variables them self or tell available for more and the variables in more details let us look at one statement and then we can warm the statement what we have is if else if else statement so notice that there is no and then so the way it works is again here, we start with the statement if or else and then in the parentheses we specify the argument of which could be like a either equality inequality of some logic function.
You will see like what can was a function that you can use here but again the way it works is condition if it evaluates to a 1 and 1 in Perl is what is called true and then 0 is false if it evaluates to 1 then you can people order is between these two brackets so this code will get executed if this one is not evaluates to a 0 then it goes into this in this lobe actually like the other panel this is what is missing.
So you think of this as more condition skill so that using the parentheses it evaluates and finds up it will gain it if it evaluates to 0 or 1 if this happens to be a 0 again then it switches and goes to this else phone you look at here basically there is no else so this is the catch-all condition there

any anything that failed to agree cute in these four buckets will get executed at least here so this is like a small example of an if statement we will go into more details and more complex examples.
(Refer Slide Time: 37:04)



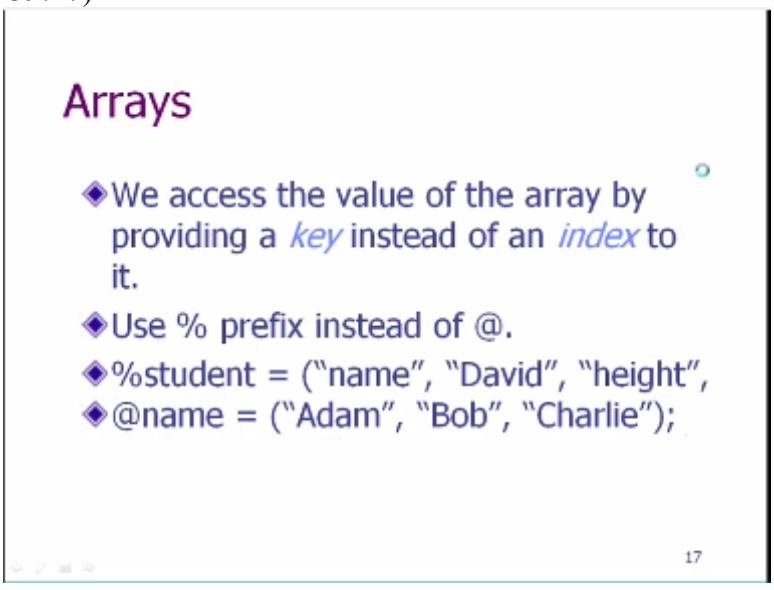
So let us look at the operators that I mentioned this the operations can be performed in two ways one is a numerical operator which is perform on numbers is like is 1 > 2 one < two things like that where both sides evaluate number then we can actually compare them with these operators so the numerical operators are = =< > <= >= != so look at that actually for the = testing is using2 = if you do only one equal to basically an assignment statement along A logical extinction similarly at the string level actually you can do an EQ < > <= >= != so this is again another way to express that information.
(Refer Slide Time: 38:04)



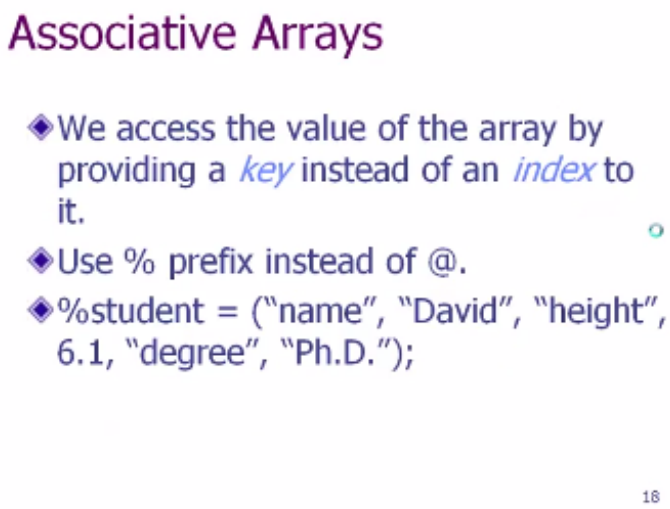
So here it is a little bit more complex complicated example with the things that we learn which is self if else--if ends so here also it starts with same things if you get the user got a simple we would not go into that mapping that became for this but here this portion is pretty much the same as the previous example which is like if you think your name then it waits for the standard in and now if the name.

So let us look at the remaining stuff as to what is going on click the $ name equal to particulars and we hear it stephen then it will print welcome Stephen if the name is equal to Jessica then it will print how are you Jessica and then it itself in place it is none of the above then it will print here what are you doing here maybe like I mean this is not your computer and more you have doing anything.

(Refer Slide Time: 39:27)



So that is the another way that giving this so now we go into the arrays so are available as I mentioned you listen to or it is basically is a collection of scalar variables and array variables typically begin with an ampersand as opposed to a $ or the follow the scalar variables so again we know that actually the Pearl is case-sensitive now we are adding one more essentially like I mean is the same command exiting.
They are actually two different among two different variables per exam here the $ name you here the dollar name and at dollar name they are two different variables they have different meanings and they can be completely different variables so a small example here which is add name = Adam Bob and tongue then ; so here you can see the basic move this is the name or the array name and then this is the array declaration.
So Adam Bob and come so let us look at how to retrieve mystery array okay before so we will figure out negative this when you do the actual program itself.
(Refer Slide Time: 41:20)



So let us look at associative array this is another key area that that will be used in a lot of times so associative array is kind of array which does not have integers or numbers as index it can have actually different in the indices within we know that it can be just a number or numerical value or any string anything can be used as a as a key or actually a as if you are the index or that array.
So again one thing is for this type of arrays you have a key instead of the actual index we do not call it as an index follicle key and then basically it need not be numerical so those are the two things that I want you to remember now let us see like what happens so it uses a prefix % or the associative array instead of this add symbol and then so the simple array could be this % student = named David 56.1 degree PhD.

So the way to read it is basically the first one is the index the second one is the valley or will be one calls it index is P and then the second one is none so in this case essentially making a deposit a student or the student array consists of three field first field is game second field is the height and third field is the degree and then it just stick assigns this values to that particular location.
So if you say like student name and value that is that we give you the output as David program do not even go there so, can also liquid register maximum wave without you and same person paint student you then pay attention and this will be given as of the is height again.
Same thing right then it will come up with explained this is degree then it will print PhD again this is a very useful concept essentially.
They told you like I mean this is used by many programs and many tools essentially for getting some more clarity also like the doing some difficult algorithm for example you can move right a code for finding the timing violations the content timing violations in circuit and you have a full every report possible and then you can write a small specific V to it and then find the 10 first unique path violation from the file.
This is a basically you need to understand the Pearl and they like what you seen technical publishing the various timing path and then we need to go and do some more massaging of that in order to get that in functional and we will actually like looking into this associative array in a big way because this is one of the key concepts.
That you should have so in C and other programming languages there is a concept of # arrays this is very similar to that concept is in think of it even though the # are dynamic # you can say like a file from various from the data structure to data so, so I think like I mean I am going to leave you with this for today because what we will be starting the next class will be mostly like how to do programming itself.
Using these data structures so just to recap we went through this variables so today as I mentioned basically we will be talking about the scalar variable so we first talked about the regular scalar variables with the column name so any scalar variables begins with the dollar and then followed by some characters and this character is case-sensitive there are a couple of ways of writing this for the variable name.
One is the basically like every word is separated with an underscore so and then make those words meaningful of what it is not in the program or you can also you try to use camel case which is basically capitalize the second word on words first word alone stays as lower case but or the second word first letter will be application then followed by so camel patients of camel case so you can use that too as a variable name.

I will let you choose because some people who like to actually do the underscores name which is which was very prevalent in 80s and 90s okay to thousands but then with the advent of the other programming languages and people when they familiar with that like Python and things like Python and take a listen to me they started writing it in camel case so I will let you choose whichever format that you want to use but just be aware.
That once you choose a format pretty much it will stick with you for equal life so I do not know it just happens so it very hard to change in the future.
And so I choose whichever one that you want but be consistent that is so that is on the scalar side then on the array variables because it actually the array variables are starts with an @ symbol it is a instead and $ name and add name are completely like two different variables and in the dollar act name the way that we specifies we write out the all the values that particular area can take into that in a straight line.
In this example we are using like a you are using names of strings as an example but this would be just difficult language that is also possible and here like string you can see that the " also " " you and then finally we saw the Associated thing so the river is a type of a data structure which gives like more makes it more versatile for call impact one of the concepts is essentially once you know associative arrays and regular expression.
Which is an in that thing that we will talk about you are pretty much covered for all then you can actually write some really complex programs by just understanding these two sets okay so yes associated array and the regular expressions, so let us see so for the one thing about this associative array as I mentioned both in array it has the key value pair essentially like that is what we call it all this particular unit , this unit is together and they are called the team and then name so you know key together.
So and then the key can be like anything this with new directives and just some strings however you want to build it you can put it like there is no one way to do it , so I think let us conclude at this point now here we can pick it up in the next class from this point onwards okay thanks.