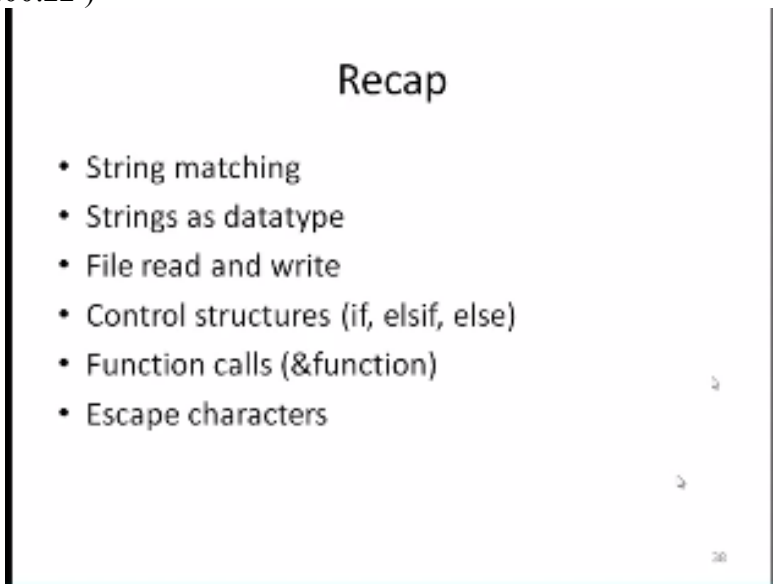
SEER AKADEMI
LINUX PROGRAMMING
AND SCRIPTING -
PERL 3

Once again welcome to this the next class on the Linux programming and scripting today we will be continuing the Perl programming language so before I go into today is topics let me just recap a little bit as to what we have so far.

(Refer Slide Time:00:22)



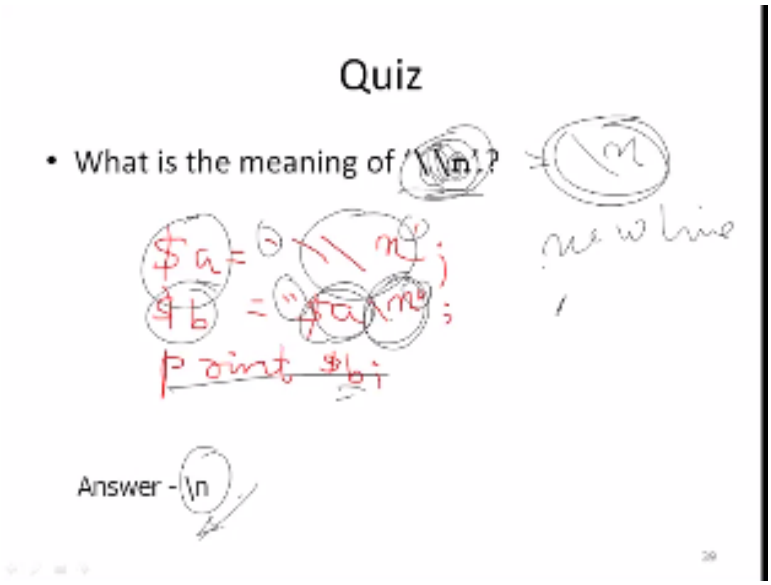
In the first lecture of Perl we started by actually looking into some basic language constructs we wrote programs to fight like the low world we also kind of understood how we reads the files of people reads input from the terminal and then outputs into the terminal we also understood a little bit about the basic data types that is a scalar data type and then an array data type array also we introduced something called The Associated associative array.

And essentially at that time we said that we see okay or string is another type of an array and then basically your string is again an array of characters and that is why it is like and with a very and then in the last lecture we learnt a little bit more about strings as a data type basically we learn that we can have strings of any length the entire book can be considered as one variable string variable and then we also did some string matching.

I hope you remember s/ and then basically the first matching that we learned was equal to followed by the tilde of the quickly character that is on one of the keys in the shift keys.

In the keyboard then that quick character along with that we said basically you add an S then we can do substitute essentially we can not only match but also we can substitute a character or another factor then we also did basically some other types of matching.

One is the transliteration that we talked about it is represented as TR and then here essentially we can change the characters from one case to another case or substitute for some characters with some other characters things like that there lot of other types of our disciplines that can be done then we also talked about how to open a file and then read a file and then how to close a file we also talked about how to write info file.
Which is using the ampersand and then we introduced the basic concept of what is the file handle essentially which is essentially the pointer to the file using which pearl actually writes or understands the concept of a file so we also know that there are specific files that are already built in functions which are the <STDIN > or < STDIN >out later than
And then you also saw the left and move standard error STDERR and > those are all for the predefined file handles which is which are essentially used to read from and write into the terminal or essentially of the program so other than that if you are defining a particular file handle we need to first open that file and assign a name or the file handle and then once we assign the name then we can use that.
So that is the so that thing that we learnt about that and then we also actually saw some control structure so that the control structure so that the control structure we learned was basically no more if then else if else if else basically that is the if then else construct and we learnt a little bit about how to write those things and then we also started talking about the function calls the function calls essentially for the function calls essentially this is something that basically.
We know that a function calls can be specified basically like a sub and then the function name and then we can also specify some argument and then once we write the function and then we can call that function into the in program by using the ampersand function then we learned about some escape characters so these are mainly for escaping special functions and here actually like the kind of distinguish between the string with the like just the “.
As well as the “” so the key difference is that inside this one anything that you write is just literally translated or will literally kept whereas here you can have additional warm variables and then variables will replaced at the runtime but program and then we also talked about a small quiz which is what I wanted to talk about in the first thing.
(Refer Slide Time: 06:50)



I actually asked you what is the meaning of this the \ \ n as you know like the \ \ n is basically the escaped characters N so within the code if you specify \ more force and more happens so I also ask you to actually try to run this program basically the $a=\\n $b=””$a\n what happens so I hope you tried this out if you have not you can still try it out but I am going to give you the answer essentially today.

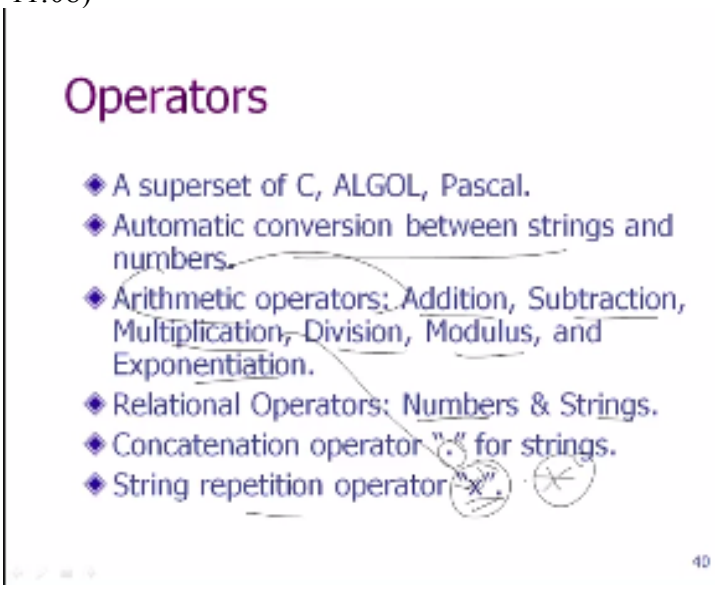
So the answer is actually or printing the $ b is the \ n basically this is also = \ so as you know like the \n we studied in this side basically this specifying \ n means that it is a newline character but in this case since we are escaping the \ it is basically this is treated as a literal and then in this is trick literally in the any positive opening so as you know like I mean within the “ only the \ and the tick or the , are the only two characters.

That can be it and it happens to be one of them so this is just a literal \ n so even if you just print this out it will \n even if you are using it within this variable and then you did it out it is only \n the second \ n since we think that as the new line so the program goes to them out of the cursor goes to the next one I hope this is clear you have any other doubts please feel free to email me or TA who should be able to answer these and questions.

So with that we will start talking about today is topic essentially today we will be continuing our discussion on the various the , will continuing from the escape characters we will go into more operators today and then we will also look at some of the in available start talking about the scalar variables unity of the scalar variables in the couple of lectures ago and today we will go more deeper into that scalar variable.

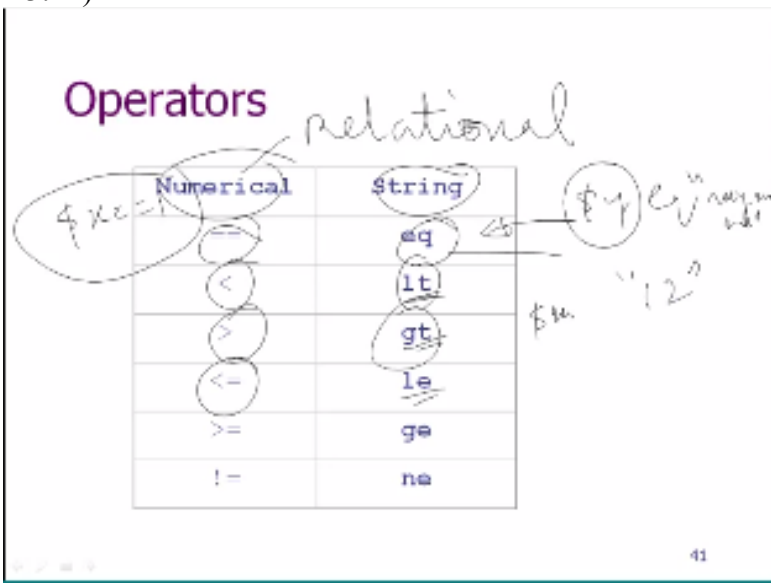
As I mentioned the goals for this course is we will start slowly building things so that by I mean we are not going to be like in the we will cover some of the basic information and things.

Like that but it will start building this programming knowledge right from the get-go I want you to study all the basic stuff and basically with every topic one by one and then at the end you can put together a data form that is not the internal source.
As you go through the course you will be able to program smaller bits of distance pieces of it is essentially even the goal call for somebody who takes both its intuitive so that you can start coding even you do not have to go to the course of many days before you start coding basically as you learn we build small scripts and then we will build it up and then finally like and by end of the course you will be able to do whatever programs that you want very easily with pearl.
(Refer Slide Time: 11:08)

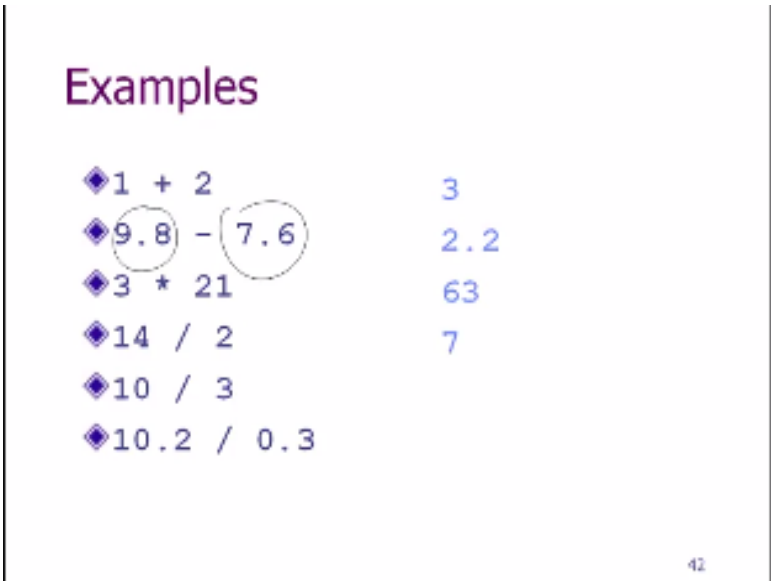


So let us look at some of the operators well so the operators in Perl is a superset of the other languages without their the C and ALGOL, Pascal basically it also provides automatic conversion between the strings and numbers there are arithmetic operators so operators are not limited to just this arithmetic operators but the arithmetic operators are essentially + - multiplication division modulus and exponentiation.
So we will study these operations form and how to use these operators in this section there are also like a relational operators or number open spring they are both are separate one if you have a number data then you use one type of relational operator and of strings use a different type of relation model base there is also a concept of this concatenation operator concatenation means joining strings So if you have like many strings that you want put together there is a. operator and then the strings they also have a repeat repetition of register just the X.
Which is very similar to the multiplication operator here the repetition operator essentially like works on the strings and basically produces multiple copies of the screen we will see like them in

how this and you know the regular multiplication operator is this extra So * symbol which is shifted in keyboard so we will see like some of the examples as to how to use them.
(Refer Slide Time: 13:14)

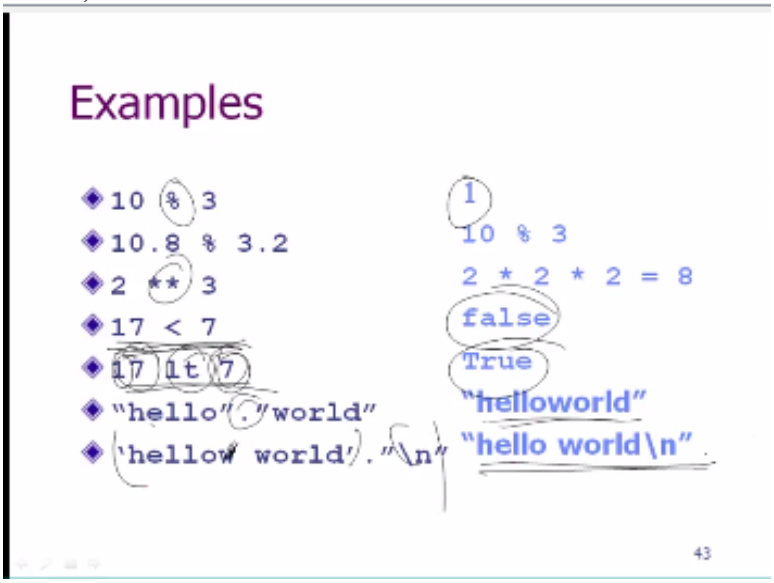


And before that let me talk about the relational operators this is the table that we saw previously in I think couple of classes ago I want to reach it once again so in the relational operations we have set up one for numbers and one for string so this is the number one number relational , and then this is for spring the national so equal to is straightforward this is to compare for EQ it is basically make two strings whether they are equal.
This one is like it is two numbers that are equal so this can be two number basically can be variable so you can say like follow X =1 this compares that if X has value is 1 and here like you can also say that Y, EQ my name underneath this so that this compare to this it is developed with no game or something so this is the EQ operator the < symbol is for the < > is for the > and < = these are like some of the things that are provided.
You may wonder like I mean how do you look < > LE and for the string operations again the equivalent numerical value is used like so for example like you can actually have all the string which is like 12 so if it is a string operator then we will use tail again my dollar number is < literal 12 so here essentially like I mean you can you can compare a number against the literal string as well so same thing like I am in the > = and then finally like = we will use some of these things in various program coming lectures You will also have some assignments using these concepts so here are some examples.
(Refer Slide Time: 15:50)



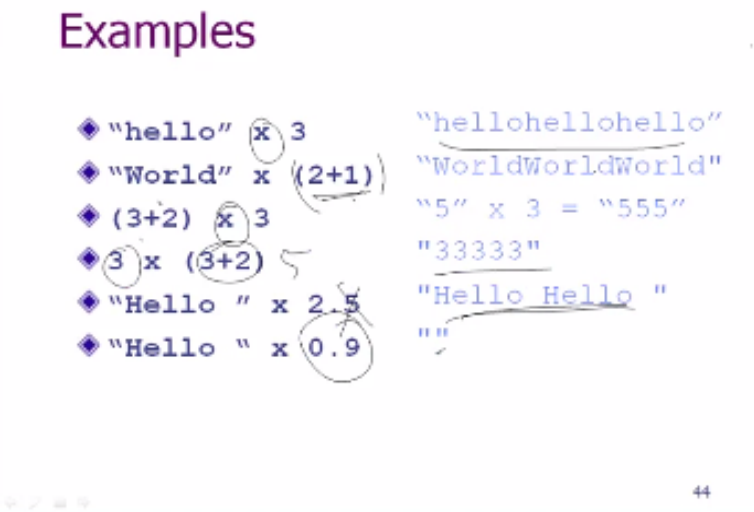
So 1+2 straightforward it is 3 this 9.8- 7.6 and the 2.2 again you see that basically you do not have to explicitly specify floating-point as you know pearl everything is this floating point 3*21 63 again easily understood 14/ 2 7 10/ 3 is boarding point number 3.333 in pearl has basically the position is limited by what your computer allows and then the 10.2 / 0.3 34 these are pretty straightforward operators sincerely pearl for addition operation - multiplication and division, so these are the basic operators now let us look at some other operators as well so this one is the modulo operator.

(Refer Slide Time: 17:22)

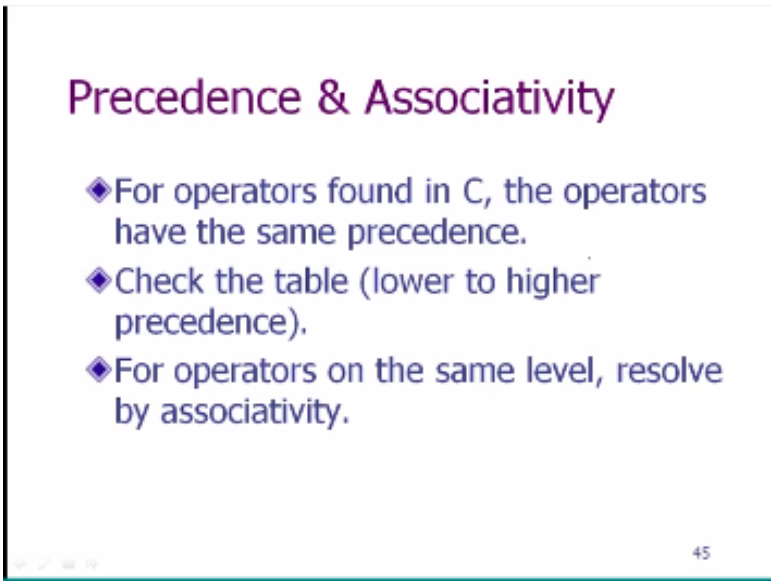


So then modulo 3 is 1 which is the remainder assumption and then 10.8 modulo 3.2 is 10 % 3 same thing and this is the exponentiation so 2^3 is 6 8 so here if trying to try to do this 17 <7 the answer is false whereas 17 < 7 in the string operation the answer is true so some of these things

basically move on based on what the operator we use is different so here are a numerical one then you use this whereas this one is used for string operation.
This is mainly because I am going to compare the first one and then it produces this answer and this is the concatenation operator them. so it produces this single one no space here this is joint and here essentially like I mean the you can look at this one as you should not have this W is a typo but you can see that actually it concatenates a little string with the newline and basically it combines it into one.
(Refer Slide Time: 18:56)



So now we look at some additional things basically like one thing that we talked about was this repetition operator which is the X operator , so the X operator essentially multiplies the hello by 3 oh you get the first output then the world X 2 + 1 + 2 + 1 is in parentheses that prints out world, world , world so the parentheses is actually operated first so 2 + 1 becomes 3 and then a times world is the mixing.
Here again when you use the X operator the 3 p+ 2 is evaluated and then that is essentially 5 and then 5 times 3 so it is 5 ,5 , 5 and 5 is treated here as a string or literal but at the same time 3 X+ 2 is actually 33333 which is basically this 3 + 2 is evaluated first which is 5 with just the piece 3 *5 times and hello times 2.5 basically like only like two times its printed out.
So the last one is the submitted and if you do it anything <1 be signal it is nothing instant about so I think this is clear you can actually play with this operators to see like I mean what various outputs are I will also give some exercises on they form to work on these operators to see how you can make use of it.
(Refer Slide Time: 21:02)

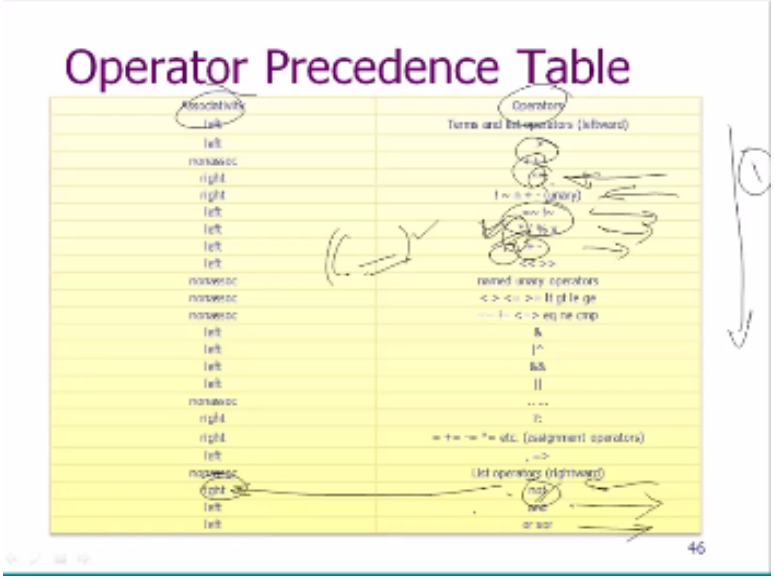


So now let us go into residence one thing that I talked about here was this parenthesis basically how parenthesis will changes buildings basically instead of world times two and then 2 +1 it is actually like the 2+1 one is evaluated first and one world same thing like here we see like some operators so it is okay so how does well understand this president are there any rules behind this how to evaluate operations.

So that is what we learn in the next section basically the rules are fairly simple essentially for all the operators found in C the operators have the same president as in C.

So this includes like +- the multiplication division etcetera and then you can check a table for the lower to higher presence I will talk about the table in the next slide and then all the operators at the same level it is resolved by associated okay so there are two ways basically one is the table which goes from top to bottom and then we also have some rules regarding the creativity which is used to resolve or the operators at the same level.

(Refer Slide Time: 22:40)



So what do I mean by you we will talk about this and if this is not clear like I mean actually there will be more chances that is covering this so here I talked about the associatively and then the operators so the president goes from top to bottom that is number one okay so if you if you see like one of them here not and then one of them here say multiply so this takes the phosphorus residence and then before this.

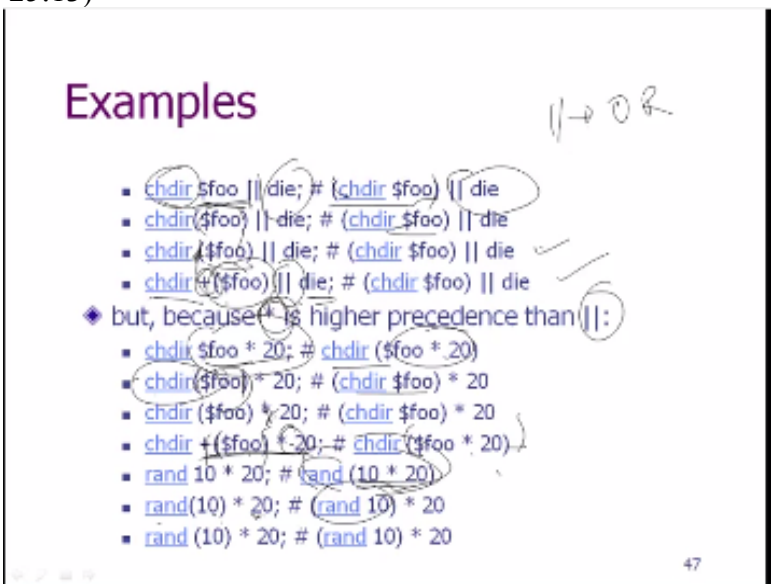
So basically if you have evaluate the multiplication before you can use but what if there are a couple of slots in the same blend line so we go back here and then we say that basically the presidency is from right to left.

So you first evaluate the rightmost one for or this one it goes this way whereas for and it is actually good and then R and XR again goes left to right and then so we can look at various things one is so there are various operators here did some of these things we have not actually like one this one we have learned which is the exponentiation which is again going from right to left so this name and then this unary operator all the same thing.

Basically whereas some of the comparison act operated both from left to right here it is again like left to right all these things like there is a , operator + - they are all going with this and then you can see that actually this + and - have lower president than this multiply and divide so if you have the + - then essentially no that will that will happen later than the multiplication and which is the normal rule anyway.

One other thing to notice basically the parentheses will change this resident order so if you put anything inside parentheses those are evaluated course and then before signing this presence table so there are all OR the operators that are mentioned here have this present so I want you to go through this I have some examples on the next slide which we will go into now.

(Refer Slide Time: 25:13)



So here are some examples so the change there is a function call CH dir and then basically war the $ foo is some variable and then we specify like I mean okay if this condition is not true then die that is the meaning of this particular statement we will talk about these kind of form statements in the next section or in the comment section but I want you to pay attention to this one so here we specify this without any residence basically we will go straight.

That is the same as writing course the CH the foo and then the or die so this and this are exactly the same if you put a parenthesis inside the FOO and then doing the same thing this that is again between this is also exactly the same there we basically like change there with no space and this exists straight away declare the $ FOO and it is argument and it is the same thing here and then finally like even if you say like change the + the $ flew FOO.

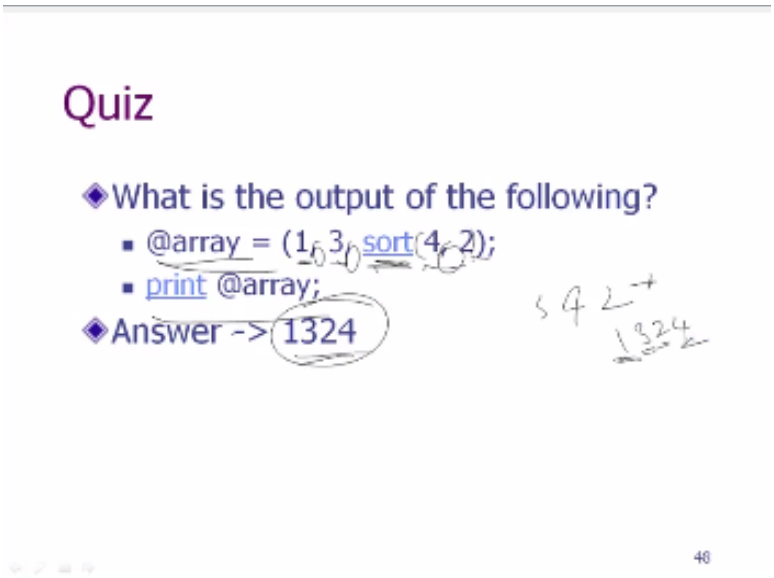
Which is an addition operator and then say like die and then again that is exactly the same that is because the addition takes residence or relational operation so this relational operation is simple move on so just in case so now when we use a multiplication operation instead of an awl or example here then this is same as the change through with $ FOO times 20 because now this one has higher precedence than relational operators and we go back to the thing.

The relational operator here that is upper/lower one then this things so again you see that basically how the present work so again when you do the CHDIR $FOO that is the same as before this so here you can see in the first statement and the second statement of this same.

Let us hear first statement second something different but the second and the third are the same because now the penthouse takes precedence over the *.

And then essentially like and it basically so that $ FOO has a suitable in there instead of playing and it is in the first statement without any parenthesis the system modifying the basically the

FOO is assigned with 20 another interesting one is this one here we put the parentheses around FOO let me add an additional operator so you can see that actually like again this has a higher precedence than +.
So it is again it is 4* 20 and that is changed then you are actually generating some , random numbers essentially so this is random 10 * 20 which is essentially random 10 * 200 I mean 10 * 20 which is 200 so it generates a random number between 0 & 4 over here and essentially you can see that actually that is all FOO steep whereas if you say like land 10 in parentheses times 20 then it generates a random number between 0 and 10.
And then that it multiplies by 20 so you can look at this one basically in a moment here there are maybe like 200 combinations and out of that one is too here you have only 10 combinations and all of that one is 2 so the answer may be like still like I am in the way in this thing this is our 0 to 200 this also has 0 to 200 but you not only like ten different values that is here you have 200 different values and it is choosing from one of them.
So this kind of presidents metrics and before this discussion will help you to write programs which you want to do it popping because in this second this section basically we want to if you want to generate a random number or some task which you think that basically which is a combination of two different types which again one more you choose to represent this way which is the correctly.
But if you are doing this way then you are actually like I mean you are not actually getting the full range to generate the random numbers because here the random you can make of almost 20 40 , 60, 80 or multiples obtained whereas here it is multiple of 1 so 1, 2 ,3 go all the way to so that is one way of looking at so depends depending on the applications you can really generate and sometimes like I mean if you do this when the reduced set even though the range for you are actually biasing it is only like in different values.
(Refer Slide Time: 31:07)



So here is one quick question again regarding the same residence that we talked about here essentially like I am declaring an array and array contains value 1 ,3 then I introduced a function on sort you can think of this sort as sorting multiple values then I give like the or 1 and 2 and then I say print $.

So what do you think will be the output of this formula so I am going to give you the answer of course for this one because may be good to actually practice these kind of problems because again this is the kind of things that you can view because it is a very flexible language so they are on like hard and fast rules on things basically we to understand how this is work so that you can write a proper program.

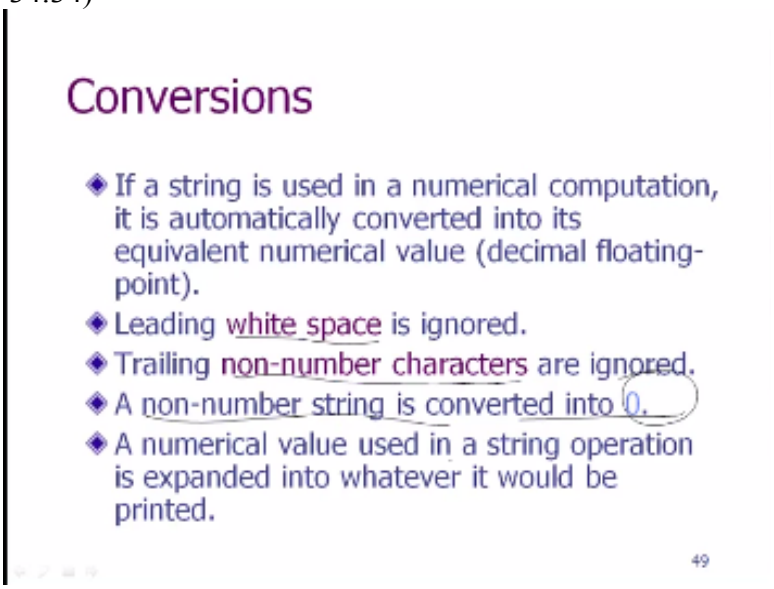
It also actually adds some additional complexity to the language and which makes debugging pretty hard because a person may have written one way he said he saw the algorithm and then if somebody else wants to debug it now they need to understand what was thinking behind the writing program before they can modify it so again it becomes very hard because for all these complex in text how do you actually go through simple again.

I want to aggregate for a program that is very clear and concise but at the same time you know the many go and appear in interviews they will ask you to the understanding to get into cases of so I think you have to know this but in practice make sure that you do not really complicate the programs and keep it simple so let us see what the output is so the answer is 1324 so here there are a couple of things that I want to note here.

Number 1 implicitly this sort is that we apply here but within that basically like them in the, operator it is the presidents so first the, is operated which is open definitely 4 and 2 and then on which the sort is operated.

So you get the result as to four then once that is performed then you have the other two , operators so the, operator says this base if you warm it context so the array is actually compares of this 1324 answer is this thing you can actually run this program and check it out whether you get the same answer or not , okay.

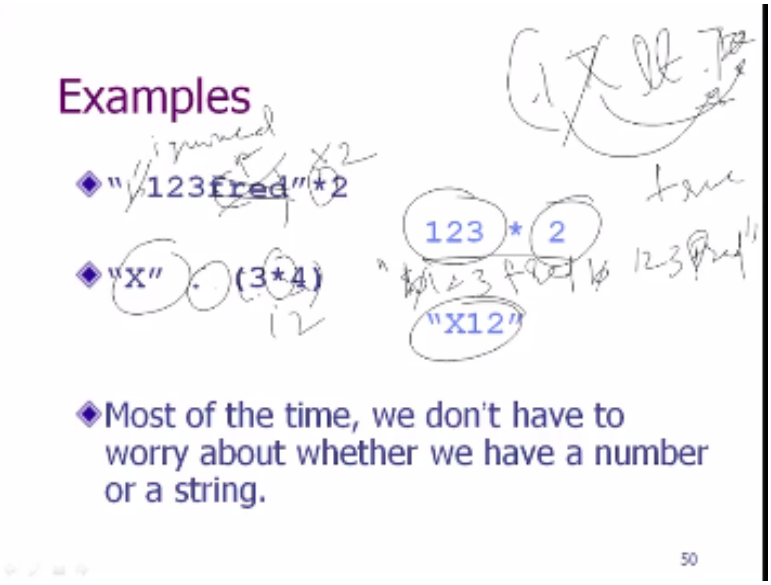
(Refer Slide Time: 34:34)



So let us move on so as I mentioned basically the operators can be operated on both strings and also the numerical numbers so the string is used in a numerical computation automatically convert converted into its equal and you will be 1 which is a decimal and the floating point value decimal floating point value so a couple of things to note one is the leading whitespace characters are ignore.

So you start with blank number and the blank is removed only the number is taken and then the filing non number characters are ignored so if it is NUMBR that is basically converted as this 0 so that is the non number string is this cannot = 0 so the numerical value is U or in the string operation its expanded into whatever it would be printed okay so let us look at some examples.

(Refer Slide Time: 35:57)



So here in “123fred”*2 what will be the output for this so this is the same as guessing 123 *10 2 because the leading this blank this is specific node and then the trialing all these non number characters are answered they are taken to their this change this is very also so the result is just123*2 and the reason why this is the true is the of this you say like I mean 123fred x2 then the answer will be like 123 Fred , and then in space okay.

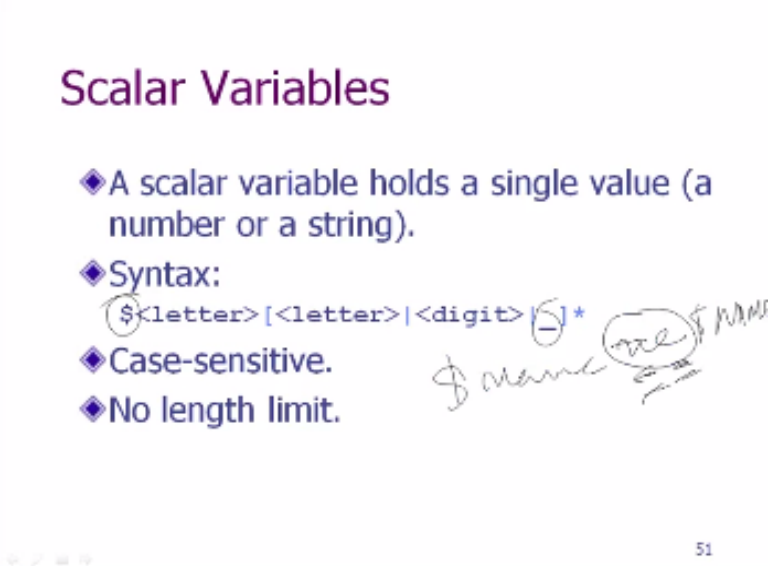
Let us look at one more example so what will be the answer for this one so as we know like I am in the Parenthesis operation is the first and so tense is performed that has the highest president along with it there is a * so this has the next president the three times fourteen three times four is a liquid first and then the answer is 12 and then that is concatenated as a string to X so you get X 12 and this is a relational operation.

So the bottom line is like I mean most of the time you do not have worry whether we have a number on the string because we just perform those operations and basically you let me in just see what is the awesome the only issue here will be when we use like relational operation buttons earlier 17 is compared with 7 actually the 17 is and >7 so that if you are saying like 17 <7 that will be false.

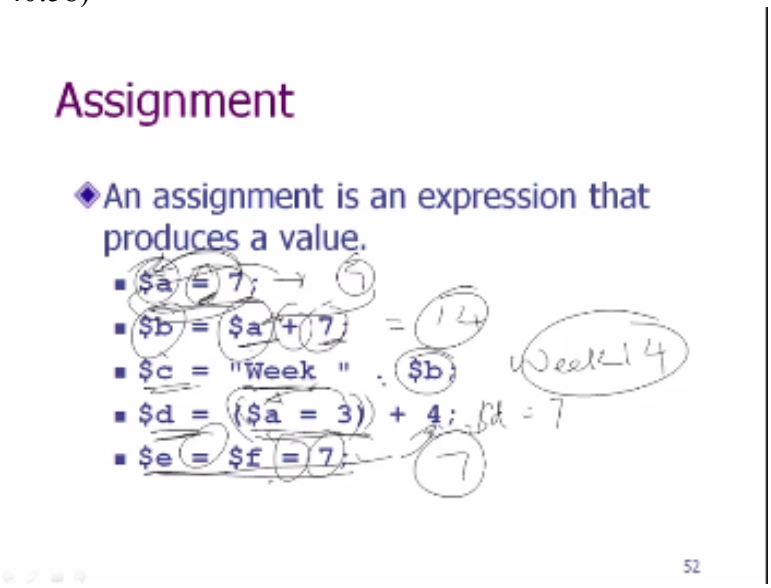
But in the string terms it is actually like one is compared against of 17 to say LT 7 now one is compared against seven first and then seven is compared against whatever this next here but here there is nothing here so some comparison is not there only the one is compared against seven and one is really <7 so this becomes a true, you so when you are writing scripts you watch out or what exactly you want the results to be because if you do something like this.

And use the wrong relations in operation then you will get a result but which you would not expect and then as a result that will cause issues and harder to debug okay.

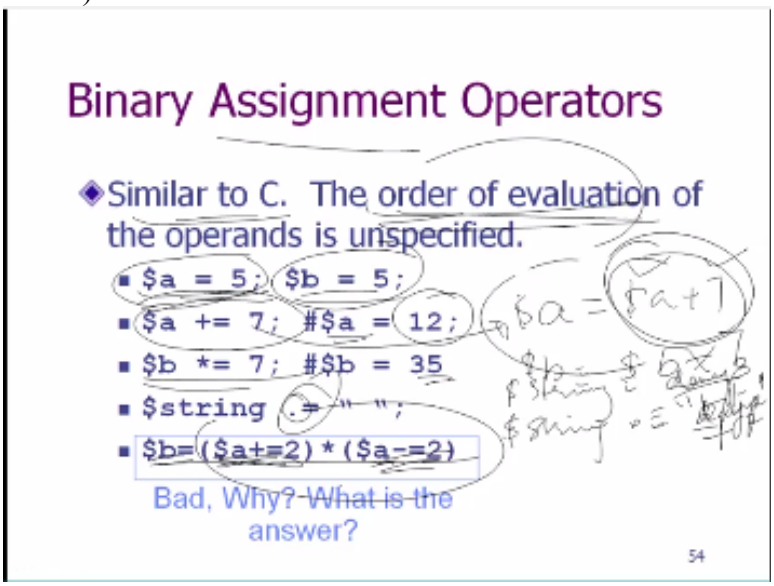
(Refer Slide Time: 39:27)



So let us also introduce form of scalar variable we actually went through this a little bit in the in the first lecture the scalar very variable holds a single value one if either a number or your strength the general syntax is essentially a letter or a digit essentially with a prefix of $ and then you can actually have an - also so other characters are not allowed in the variable and essentially like I mean for demo it basically rejects the additional variable additional characters.
And it is case sensitive so you can say like I mean same thing again a normal name is not = $ E okay this is true in both this and the nests you can have different values for this one and then for the name is that there is no length limit so you can have however you want so let us look at how we can use them basically.
(Refer Slide Time: 40:58)



One way to use the scalars is through an assignment so you can assign values to the scalar very good and which you in turn you will be using it or various things essentially so for example here a typical assignment will be $a =7 and then we will say like okay $ B=$a +7 so here you notice actually like that our that is this is the quality of inter so here we just assign a value or assignment operator where we actually assign a value to this particular variable.
This is also come sometimes which can call it as initialization where we first assign a value one and we can do some operations on it here we assign $a+7 to this one so if the boundary changes same to nine at some point then this that will be reflected here.
So currently this is = 14 but it can change based on you and then here you can see basically that gal C is week concatenated with $ D so can somebody say what will be the answer for this one it is the that is week 114 so that is the way that we can write it and then here we say like D = 3+4 so now what will be the value for the D so here essentially like I mean you can think of the parenthesis is actually change the president.
So evaluate with first and then that so in when we evaluate this is an assignment operator when three is assigned to a and then that +4 basically becomes $B is =7 here it is basically like a continuous assignment with until you one is defining a $2 F which is $7 so both of them will become same so I think this is the way that we can change the president is essentially general form we can move some operations basically on the various variables.
The scalar variable is one of the most widely used variable inputs pretty much most of the operations will tell us so the scalar variable we will learn a lot of things about the scalar variable becoming lectures.
(Refer Slide Time: 44:18)



So let me also go into the binary operators essentially or binary assignment so essentially this is very similar to C we mostly would not move which will be fairly easy The order of evaluation is unspecified so if we go back to our which is that we talked about here so the you the some of the + are they are basically very similar to this one but they do not have any kind of associative so now let us see like I want that mean.
So here a=5 and b=5 so it is a standard that is given so now we are assigning a += 7 first of all what does it mean this means actually both a follow a =+7 the reason why this order is an evaluation is unspecified because it does not matter how this the is evaluated okay so in this case the $ a will have a value of 12 because it is basically that is the value that it settles at okay so this is evaluated first this is evaluated cause it does not matter.
This is so this introduces some kind of ambiguity later on you will talk about that actually and then now we come to this basic so here again same thing back $b=$B times then that is 35 and then there is also like this concatenation operation which can always be a binary assignment which is for lot of people use this shorter for binary assignment where they with a down lo string equal to they gave and then.
Maybe follow string not equal to la offline so then now the bottom string will equal to they live something like so it is a it is basically like I mean we can actually create new strings very easily by using this binary assignment so now we come to this last example $ B is=$a +=2 and then *$2 $a-=3 so what would be the answer here first of all this assignment is the worst assignment so never do this it is bad why is it bad the reason why it is bad is because inside this whole thing this order of evaluation is unspecified.
So we do not know like I mean whether this is a valid at first of this is one so now how will you resolve this and what will be the N for I want you to actually test this out before I give you the answer because they have gained like I mean this is one of the P points of understanding that we should be able to understand how this is this is done so I am going to actually stop at this point and do a recap of what we learnt today and then we will pick it up from this meanwhile I want you to experiment with this particular assignment.
See what happens and explain why you are getting that answer so first of all find out what your answer will be how do you do this so all of this one basically $a is 5$ or actually just make a $a as 5 and then just do this operation see what you get because again this order of evaluation is unspecified what does that mean find out and explain to me in the next class or what happens and in fact I will also explain to you as to what you should have seen and why you okay.

So let us just recap what we learned today so we started with some operations we went through like the various operators essentially in Pearl operator we have a number of operators with superset of C or ALGOL, Pascal and there is automatic conversion between string and numbers which means for how we can do this and in Perl actually there are arithmetic operators addition subtraction multiplication division modulus and exponentiations.
And then we have a relational operations exponentiation which is the “” and then we have the relational operations and one set for numbers and one set of strings actually you do not have a data type where we can operate these things form you can easily operate the string relational operator on the numbers and then the number operator on strings even though like I mean when we operate the numerical relation operators.
On straight or more strings it can completely change that but if you use the string comparison for the numerical assignment then we also looked at the concatenation operator or strings then we also studied about the string of it strings repetition of this is the X before multiplying numbers then we went to all these various operators we also studied a mother residence order so this is one of the key things that we will again revisit in the next lectures.
In president essentially let being said that basically pearl you are representing this first of all it goes from top to bottom and then it also follows whatever the associate we will the first people in for the same operators you can use this just this one row so and then and then that also varies essentially for some operators before right to left and some for some operators it is left to right for example the not functionalities a right to left right.
And OP, XOR and X all right now I mean left to right so this we learnt and then we learn also lengthen power plants or some complex questions then we studied about the conversions essentially which is how can we use string in the numerical computation well.
And then how can we use for numbers or comparing in a string way actually one of the conversion thing the rules are clear one converting from string to a number the leading white spaces.
Involved and then the tiling non numerical characters sorry then whatever is remaining that is your number essentially, and then the numerical value used in the string operations expanded into whatever it would be or then it will be if then it can be printed so this is something that we also saw and then we went into the scalar variable for essentially these are scalar variables variable all the single value.
It is usually represented by $ followed by number of letters or digits essentially so you can have like $ 123 as a scalar variable involve the value you even though we do not want to we always

typically have a letter essentially starting actually here also electing syntax we start with the letter then we can have any digits so all little so or even more and wonderful so we will so the $123 is not allowed I sell the rest dollar a 123 is okay.
So again the scalar variables the variable names are a case sensitive and there is no link limit for the variable so I want you to keep in mind about these rules and this syntax issues in syntax form basically then I can go in and I will ask you to follow like one type of syntax you can so pearl allows this underscore and TP will typically write like name one or me and the score 133 or you can specify like them follow mine pearl.
As another so they use - various words to form available you can also use camel case because it is a case so sensitive and mortgage you can feel like they are mine as one more me 123 so you can choose once time of global and we believe it people to the style and stick to them for life so sleep and people who are well versed with all they typically choose this with - so I would like you to choose each other one that you are most comfortable.
Within representing various variables then we also learnt about assignments the two types of assignments one is a regular assignment which is basically assigning values we will still available we use like various operators to achieve that assignment so assignment you can think of that as an operator represented by equal to and then we use various types of performance to feed into that particular variable on left hand side.
So we also learned about this binary assignment operator which is these kind of operations essentially because to operators linked together may be also So basically one thing is the order of evaluation on both bands is unspecified for which one evaluated first is one known even though like I mean most of these kind of cases so we evaluate.
If basically you and I same as same B =7 so +6 evaluated first before that So A will get the value of 12 or B will get the value of 35 but when you use it with this kind of complicated be there this you are reducing one type here and another type with the same variable.
The value sometimes is more portable so I also gave you this example and we want this will be a quick quiz or next time to see whether we understood this one this command and see whether what the result you get share with me I will also share my result with you all next time and give you an explanation as to why I see this value.
So hopefully I think this is not there we are getting more and more people so thank you again for listening thanks a lot see you next time.