Foundations of R Software

Prof. Shalabh
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Basics of Calculations
Lecture - 16
Logical Operators

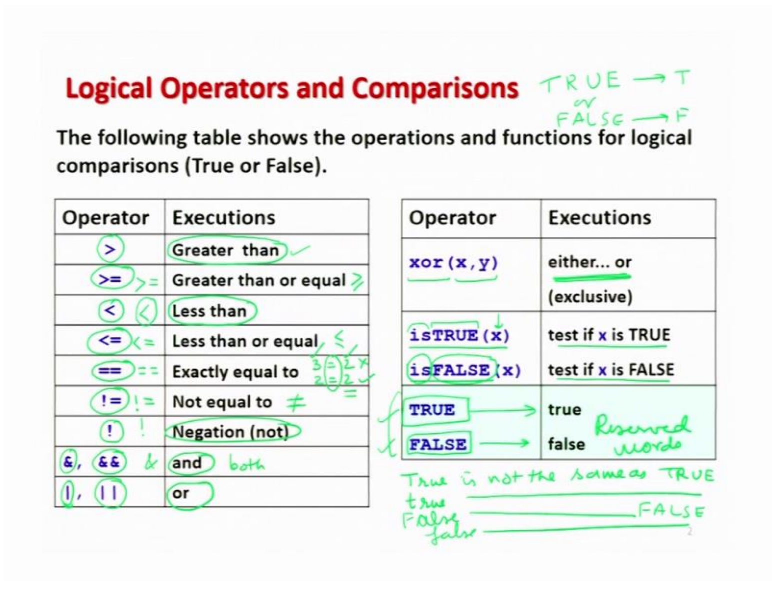
Hello friend, welcome to the course Foundations of R Software and now from this lecture we are going to begin with a new topic and this topic is Logical Operators. You know that we have two types of operators one are mathematical operators and say other are logical operator. The mathematical operator will give you a numerical value whereas the logical operator they will give you the answers in terms of whether the answer is correct or not.

And this logical operator, they are very useful actually. You will see that whenever we want to frame a query in databases they are very useful, similarly when we want to execute something under some constraints, under some conditions, then this logical operators helps us. So, the first question comes here that what are these logical operators in the R software and how they are operated? How to interpret their outcomes?

And if I try to take an example suppose if I say 8 is greater than 6, the answer is yes or no or true or false. So, that this then this greater than sign, this is my logical operator, but in case if I try to find out the difference between 8 and 6 this is 8 minus 6 which is equal to 2. So, this is. So, here minus is a mathematical operator, but when I try to see 8 minus 6 is greater than 0 or not then this greater than becomes a logical operator.

So, similarly we have different operator like less than greater than less than equal to greater than equal to not equal to and equal to equal to is not like the mathematically equal to sign, but this is logical equal to sign. So, these are some logical operators that we are going to discuss in this lecture and in some forthcoming lectures also and we will try to see that how we can manage the computations and calculation using the logical operator. So, let us begin our lecture.

(Refer Slide Time: 02:46)



Logical Operators and Comparisons

TRUE → T
or
FALSE → F

The following table shows the operations and functions for logical comparisons (True or False).

Operator	Executions
`>`	Greater than
`>=` >=	Greater than or equal ≥
`<` <	Less than
`<=` <=	Less than or equal ≤
`==` ==	Exactly equal to 3 = 2 ✗ 2 = 2 ✓
`!=` !=	Not equal to ≠
`!` !	Negation (not)
`&`, `&&` &	and both
`|`, `||`	or

Operator	Executions
`xor(x,y)`	either... or (exclusive)
`isTRUE(x)`	test if x is TRUE
`isFALSE(x)`	test if x is FALSE
`TRUE` →	true
`FALSE` →	false

Reserved words

True is not the same as TRUE
true
False FALSE
false

So, these are the logical operators and I will try to show you what does they do. Remember one thing that logical operators will always give the answers in terms of TRUE or FALSE or you can also write TRUE as T and FALSE as F in your R commands and you might recall that in an earlier lecture I had explained you about these two operators TRUE and FALSE and I had explained you that these two are the reserved words and which are our logical operator.

So, now this is the place where we are going to talk about them. So, ok. So, we have got a different types of logical operators. So, first let me try to explain you that what are these operators and what are their corresponding operators in the R software. So, when I want to compare something then we have two options whether the value is greater than or less than. So, in order to execute the greater than operation, we have the symbol here greater than sign. This is the usual sign in mathematics also when we try to compare two values.

And when we have a condition that we want to compare like greater than or equal to in mathematics, then this greater than or equal to in R software is indicated by this symbols that we write greater than sign and we also write equal to sign and similarly in case if I want to compare something like something is smaller than this, then we have this operator for it is our usual less than sign and this operates the operation of comparing for less than.

And when we are trying to compare with respect to less than or equal to, then again I try to write here less than sign and I write here equalty sign like this in the R software that is just like greater than equal to we have here less than or equal to. Now when we try to write something as 3 equal to 2, this is wrong, but if I try to write down here 2 equal to 2 this is correct.

So, here what are we trying to do? We are actually trying to compare the numbers on the left hand right hand side of this equality sign. So, actually, here we are trying to indicate that these operators, which I have used here as equality sign, these are our mathematical operator, but here we mean to use them as they are logical operators.

So, the logical operators for this equality is two equality signs like this and this. So, this will indicate whether the two values are exactly equal to and in case if I want to compare like not equal to which is our like this in mathematics indicated by this symbol. So, this operation is done by this exclamation sign and equality sign like this. And when we just want to do that negation; that means, not then we have here a symbol exclamation sign that is a symbol on your keyboard.

After this when I want to join two such operations then I have two options one option is that I am trying to say that if this condition and this condition. So, when I want to join two logical statement through and means both then we have here symbol ampersand and like this one, but there are here one symbol and two symbols. So, what is the difference between using this ampersand and one times or two times, that I will try to explain you.

And similarly when I want to make a logical comparison with respect to the statement or for example, this happens or this happens then, in that case our symbol is this vertical line. This is also a symbol which is available on your keyboard. So, we use here single line or double line, but what is the difference between single and double line that I will try to show you with some examples. And similarly if I have two statements x and y and I want to say operate like either x should be operated or y should be operated.

So, something like either or operations, then for this thing we have a command here xor and inside the parenthesis we write x comma y means either x is operated or y is operated. Now in case if I have some statement and I want to know whether this statement is TRUE or FALSE then we have one more command here which is like isTRUE, but if you try to see here how it is written i s this is in small letters lowercase alphabets and TRUE is written in the upper case alphabets.

And whatever I want to compare that I have to give it inside the parenthesis, suppose my statement here is x and I want to know whether this statement is TRUE. So, this command is TRUE will test if x is TRUE and similarly if I want to test if x is FALSE then similarly we have a command here is FALSE, but once again you have to be very careful that i s is in the lower case alphabets and F A L S E this is in the upper case alphabets.

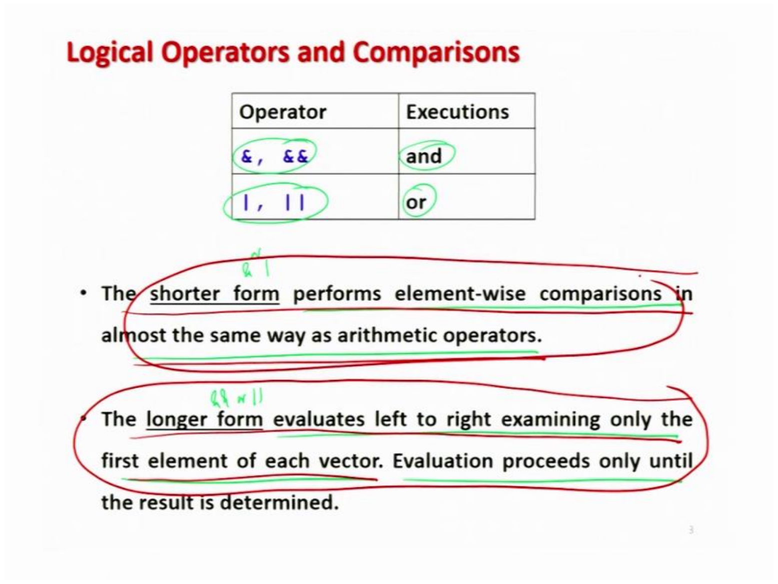
And these are the conditions which are quite needed whenever we are trying to do the programming that we want to say for example, if this condition is TRUE then you try to operate this statement and if this condition is FALSE, then you try to operate with this condition, right. After this as you already have talked about it, we have the two logical operators which are TRUE and FALSE. So, capital T capital R capital U capital E will indicate the statement is TRUE and capital F capital A capital L capital S capital E FALSE that will indicate the statement is FALSE.

And I would like to have your attention that when you are trying to write down here like as T r u e means T is capital and all other alphabets are in the lower case then this is not the same as TRUE. This TRUE and FALSE in upper case alphabets, they are the reserved words means you cannot use them in any other variable means you cannot give a variable a name like as TRUE or FALSE, but in case if you think the pattern means either one or more than one letters are not in the upper case alphabets in TRUE and FALSE then it is different.

So, similarly if you try to write down here all the alphabets in lower case in TRUE, then this is also not the same as TRUE. And similarly for the FALSE also if you try to write down here False like this means F is capital and a l s e is in lower case then it is also not the same as here FALSE. And similarly if you try to write down here false like this all in lower case alphabets this is also not the same as FALSE which is a reserved word. So, that is what you have to keep in mind, yeah.

I am sure that at this moment you might be confused that what are these operations and how would they are trying to work, but I promise you as soon as I take some examples means everything is going to be 100 percent clear.

(Refer Slide Time: 11:51)



Logical Operators and Comparisons

Operator	Executions
&, &&	and
|, ||	or

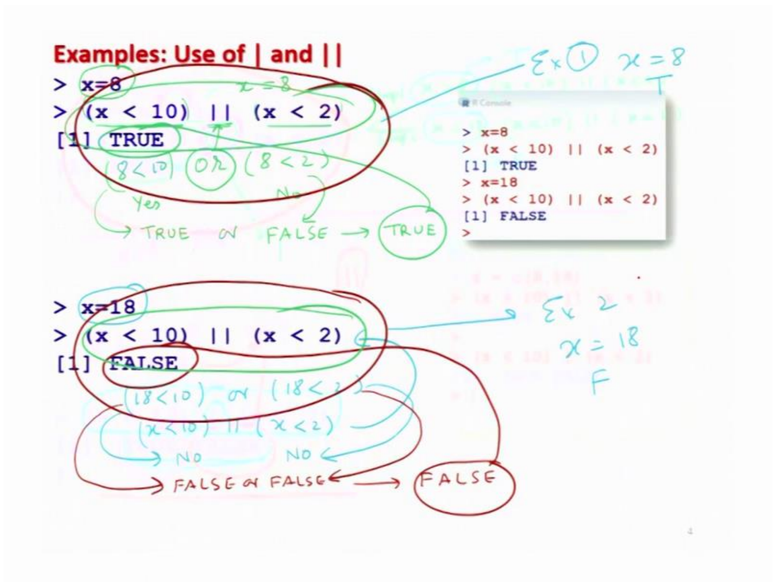
- The shorter form performs element-wise comparisons in almost the same way as arithmetic operators.
- The longer form evaluates left to right examining only the first element of each vector. Evaluation proceeds only until the result is determined.

So, let us try to take some examples, but before that as I said that we have two types of operators means using ampersand and this vertical lines which are used for and or and then we have an option that either I use 1 symbol or II symbols then these two have a different interpretation.

At this moment I will simply try to make the statement and later on towards the end of this lecture I will try to show you with some example then this statements will become clear. So, this shorter form; that means, you are using only the one symbol this or this this performs the element wise comparison in almost the same way as arithmetic operators whereas, the longer form; that means, two symbols like this or this.

They evaluate left to right examining only, the first element of the each vector and evaluation proceeds only until the result is determined. So, you will see that when you are trying to deal with more than one values in a data vector then these operations will make a difference.

(Refer Slide Time: 13:07)



Examples: Use of | and ||

```
> x=8
> (x < 10) || (x < 2)
[1] TRUE
```

```
> x=18
> (x < 10) || (x < 2)
[1] FALSE
```

```
> x=8
> (x < 10) || (x < 2)
[1] TRUE
> x=18
> (x < 10) || (x < 2)
[1] FALSE
>
```

And what is the difference I will try to show you with some examples, but let us first try to begin from scratch and we try to understand what is the meaning and utility of these logical operators; so, ok.

So, now let me take here one example, suppose I choose here a value x equal to 8; that means, you know what is the meaning of this? That the value 8 is assigned to a variable x. Now, I want to choose that whether this value 8 is smaller than 10 or this 8 is less than 2. So, for this or I have this symbol here and I try to write this statement as a x less than 10, 2 vertical lines and then x less than 2 and for clarity I have written them inside the parentheses.

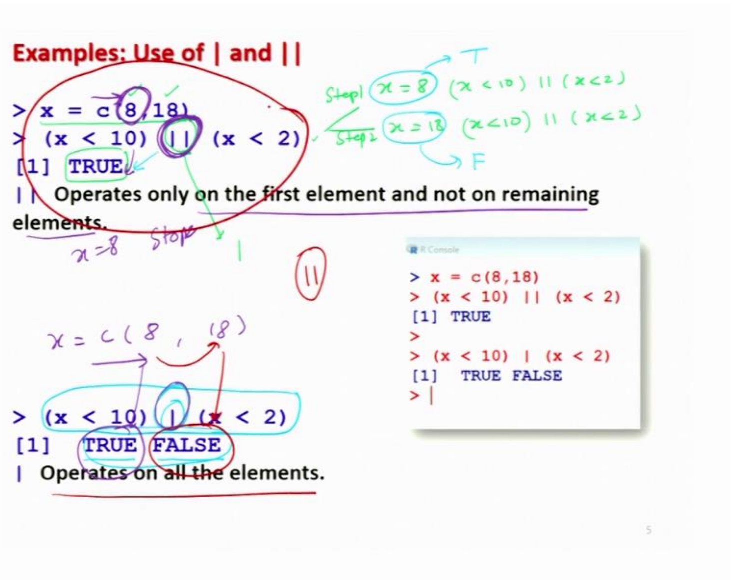
Now let us try to first understand what happens. Do you think that 8 is a smaller than 10? Answer is yes. Do you think that 8 is a smaller than 2? Answer is no. So, that means, when I am trying to say 8 is less than 10 this statement is TRUE and when I am trying to say here 8 smaller than 2 then this is here FALSE. So, and I want to obtain the answer if any of the condition is TRUE.

So, TRUE or FALSE will give you here TRUE because one of the condition is TRUE right. And I will try to take here more examples so, that this concept become clear and that is why this TRUE is coming here in this outcome. Now I try to take here one more value of x and I try to take here x equal to 18. So, now, I try to use here the same condition and I want to test here whether 18 is smaller than 10 or 18 is smaller than 2.

So, I try to write this statement here let us say 18 is less than 10 x is the value of 18. So, I simply write here x less than 10 and in place of or I use the logical operator and then I try to write down here x less than 2. So, this is exactly it is written here. Now tell me whether 18 is smaller than 10 TRUE or FALSE? The answer is no it is not TRUE and similarly if you try to compare here whether 18 is smaller than 2? Answer is no, this is not correct.

So; that means, when I am trying to compare 18 less than 10, then this statement gives me an answer FALSE and when I am trying to compare 18 is smaller than 2, then it is giving me an answer FALSE. So, when I am trying to say here, trying to combine these two statements FALSE or FALSE then the answer is going to be FALSE, means either of this statement is correct. So, both the statements are FALSE. So, this answer will come out to be here FALSE and this FALSE is given here, right.

(Refer Slide Time: 17:10)



Examples: Use of | and ||

```
> x = c(8,18)
> (x < 10) || (x < 2)
[1] TRUE
```

|| Operates only on the first element and not on remaining elements.

```
> (x < 10) | (x < 2)
[1]  TRUE FALSE
```

| Operates on all the elements.

```
> x = c(8,18)
> (x < 10) || (x < 2)
[1] TRUE
>
> (x < 10) | (x < 2)
[1]  TRUE FALSE
> |
```

So, now, let me try to take here one more example and try to show you when you are trying to use one symbol and two symbols for this R operator. So, now, I try to take here two values say x equal to 8 and x equal to 18 and I try to put it inside the data vector and

I try to use the same condition whatever I have this condition that x is less than 10 and x is less than 2 the same condition I try to implement here.

What do you want and what do you expect? You want R to operate like this that first of all, it will try to choose the value here x equal to 8 and then it will try to verify what is the answer of this command that x is less than 10 or x is less than 2 and after this, then in the second step, step 2 you want that x should pick up the value 18 and it should try to test whether x is less than 10 or x is less than 2, this is what you want to do.

So, now in order to do this thing, you try to operate here only the or operator and you use here two vertical lines. So, now, let us try to see what happens. The answer comes out to be only here TRUE and now I try to replace these two vertical lines by one vertical line this is also your or operator and try to operate it. Now if you try to see here, here I am trying to use the same condition, but with only one operator and the answer comes out to be here like this TRUE and then FALSE.

Now, what is happening? If you try to see in this earlier slide this is your example 1 and this is your here example 2. Example 1 is trying to consider x equal to 8 and example 2 is trying to consider x equal to 18 and their answers are coming out to be in the example 1 answer is TRUE and then the example 2 answer is FALSE.

So, when you try to consider here this example 1 at x is equal to 8 the answer should come out to be here TRUE and when you are trying to consider x equal to 18 the answer should come out to be here FALSE, but when you are trying to use 2 vertical symbols then the answer is coming out to be here TRUE and when you are trying to consider here only one symbol then both TRUE and FALSE are coming out here.

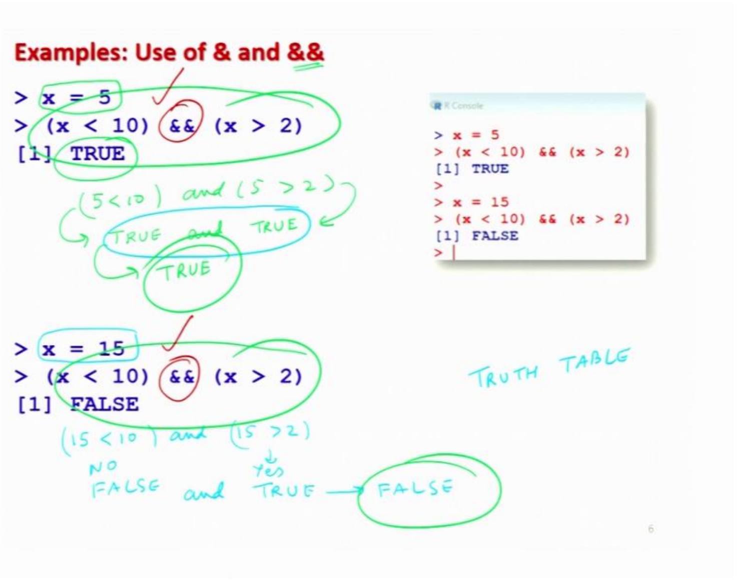
So, what is really happening? Now this is what exactly you have to understand here and the same operations and same type of logic will be happening at other values also. So, what is happening here that R starts functioning and when it finds that there are two symbols two vertical lines then it starts taking the first value and it takes the first value that is x equal to 8 and then as soon as it gets the answer TRUE after this, it stops.

So, this double symbol is operating only on the first element of this data vector and it is not moving to the remaining elements. On the other hand, when you are trying to use only the single operator; that means, only one vertical line then what is happening that you are trying to write down here x is equal to c 8 and then 18. So, R starts working, R comes to first here at this 8 and then it finds the answer as here TRUE.

And then R moves further to the next value in the data vector and then it finds the correct value, which is reported here as a FALSE. So, this single vertical line this is going to operate on all the elements in the data vector and that is the difference between using the single line and double line. And if you try to read this sentence what I shown you here now you will understand it the shorter form performs element-wise comparison in almost the same way as arithmetic operator.

The longer form evaluates left to right examining only the first element of each vector. So, when you are trying to use here the, this longer form is like 2 lines then it is evaluating only the first element in the data vector that is what I wanted to say. So, as I promised that as soon as I take an example the things become more clear. So, now, you can understand here that how R is working with this logical operators.

(Refer Slide Time: 22:36)



Examples: Use of & and &&

```
> x = 5
> (x < 10) && (x > 2)
[1] TRUE
```

```
> x = 15
> (x < 10) && (x > 2)
[1] FALSE
```

```
> x = 5
> (x < 10) && (x > 2)
[1] TRUE
>
> x = 15
> (x < 10) && (x > 2)
[1] FALSE
>
```

So, now after this I try to take here one more example and I used here the and operator right. So, first I try to take here two and operators and then try to show you, but before that I have here one question for you. When you were trying to use here the two vertical lines here and when you were trying to use two vertical lines here in this earlier example then why it was giving you the correct answer?

The answer is very simple that in the earlier example, you were trying to consider only one value that is why you could not find whether R is creating any trouble or not, but as soon as you came to the situation of a data vector and you had more than one values then you found that this double symbol is going to create a problem. So, now, the similar story is going to happen with this & operator also, but let us try to see what happens when we try to use this single and double & operators in the R console with some data vector.

So, once again I will try to take here some examples and try to explain you. So, let me try to take here the first value here x equal to 5. So, now, this x has been assigned the value 5 and I try to test here a condition whether 5 is less than 10 and 5 is greater than 2. So, now, if you try to see the answer here, 5 is smaller than 10, answer is TRUE, yes, it is TRUE and then 5 is greater than 2, answer is yes, this is TRUE.

And now you are trying to join this TRUE and TRUE. So, when both the statements are TRUE then TRUE and TRUE is going to be here TRUE and that is what is mentioned here. So, both the conditions are satisfied only then the condition is going to be TRUE, right means sometimes you have seen that whenever the grading is done in the examinations and the final results is declared for example, if you have two subjects, say subject 1 and subject 2.

And in case if you have a condition that how are you going to be declared that you have passed the examination? I have here two options I say you have to pass in both the subjects and second option is that you have to pass in either of the subject. So, when I say that the candidate has to pass in both the subjects; that means, the candidate has to pass in subject 1 and the candidate has to pass in subject 2 only then the candidate is going to be declared as passed.

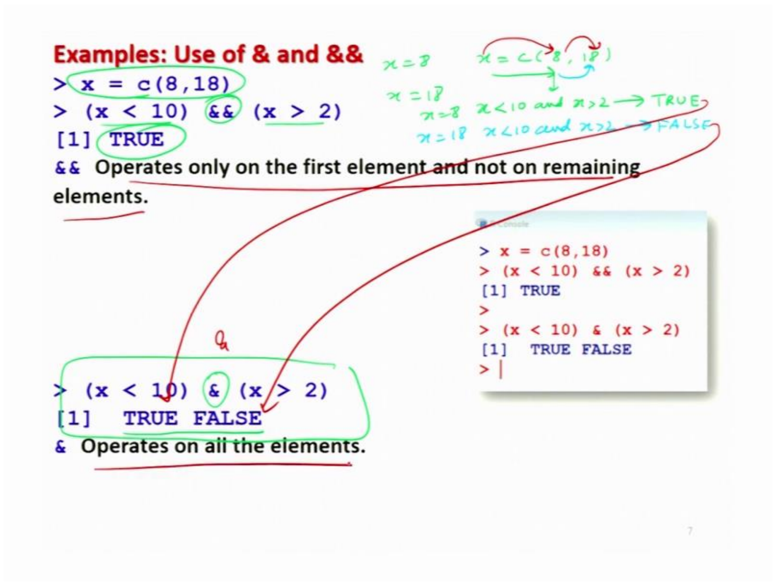
The other condition is, I am trying to say in case if the student pass the subject 1, but suppose unfortunately the student fails in the subject 2 or vice versa that the student fails in the subject 1 or pass the subject 2 then in that case the student has passed either of the subject either 1 or 2, subject 1 or subject 2 then in this case the student is declared to be passed.

So, these are the conditions that we are trying to test here using these two operators. So, now, I try to take here is another example I try to take it here x equal to 15 and then I want to here test whether 15 is smaller than 10 and 15 is greater than 2. So, these are my two conditions which I want to test. So, in case if I say 15 is smaller than 10 this is no this is FALSE.

When I am trying to test a condition 15 is greater than 2, answer is yes, and this condition is TRUE and when I am trying to combine it with here and then one condition is FALSE and say another condition is TRUE. So, this is going to give you the answer FALSE why? Because it will give you TRUE only when both the results are giving you the answer TRUE, which happened in the case number 1, right.

In this case, even if the second condition is also FALSE; that means, the outcome of suppose both the outcome is FALSE and FALSE even in that case this will give you an answer FALSE. Some of you might recall that there is a concept of truth table, right and that table gives you these types of outcomes whether TRUE and TRUE is TRUE or TRUE and FALSE is FALSE. So, I will try to discuss it little later on, but first I want to explain you what is happening, right.

(Refer Slide Time: 28:00)



Examples: Use of & and &&

```
> x = c(8,18)
> (x < 10) && (x > 2)
[1] TRUE
```

&& Operates only on the first element and not on remaining elements.

```
> (x < 10) & (x > 2)
[1]  TRUE  FALSE
```

& Operates on all the elements.

```
> x = c(8,18)
> (x < 10) && (x > 2)
[1] TRUE
>
> (x < 10) & (x > 2)
[1]  TRUE  FALSE
> |
```

Now I try to do the same operation which I did with the or operator with the single and double symbols in the case of and, and try to see what happens the result is going to be the same. So, since now you have understood the first operation it will not be difficult for you to understand the second operation. So, now, I try to take here two values say x equal to 8 and x equal to 18 and I try to combine them in this data vector here like this.

Now, I want to use the same condition which I have used in the earlier example. You can see here this is x less than 10 and x greater than 2 right. So, this is the same condition x less than 10 and x is greater than 2 and I try to use here single and double operators what is your objective? You want that x has two values. So, you expect that R will start working and it will try to first choose the first value and then it will try to here judge whether x is less than 10 and x is greater than 2 for x equal to 8.

And then it will try to find out the answer and you have seen that this answer is coming out to be here you can see here this is here TRUE and in the second case the answer here is FALSE. So, this answer will come out to be here TRUE and then in the next step this or will move further and it will try to pick up the second value in the data vector and then with this x equal to 18, it will try to test the same condition x less than 10 and x greater than 2 and you have seen that this condition is giving you the answer FALSE.

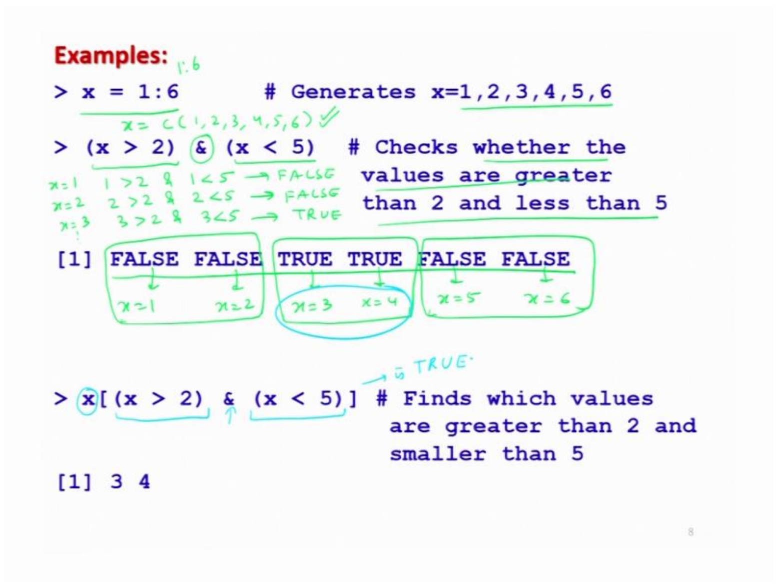
But what do you observe? If you try to operate this thing on the R console here and if you try to use here two & operators the answer is coming out to be here TRUE only the first one. On the other hand, in case if you try to choose the same x and try to operate the same condition with the single & here then the answer is coming out to be here TRUE and FALSE you can see here.

So, now, what is happening that when we are trying to work with single & then R starts from the first value in the data vector x it takes the answer TRUE and it brings it here and after that then R moves to the second value it finds the answer and it brings here as FALSE. So, the same thing is happening that when you are trying to use double operators then it is operating only on the first element and not on the remaining elements.

And when you are trying to use single & operator then it is operating on all the elements, right, but once again why this problem was not there when you were trying to work here and here, because you were trying to deal only with the single value and then you are using here double & operators. So, double and operator is working only on the first value and since there is only one value. So, the one value is the first value and that is why you could not face this type of problem.

And as soon as you consider the data vector, where you have more than one values you observe this problem and once again if you try to look at this statement, which I shown you this is the same thing if you try to see I am trying to write down here that the longer form evaluates left to right examining only the first element of each vector, right. And the shorter form perform the element wise comparison in almost the same way as the arithmetic operators.

(Refer Slide Time: 32:13)



Examples:

```
> x = 1:6           # Generates x=1,2,3,4,5,6

> (x > 2) & (x < 5)  # Checks whether the
                       values are greater
                       than 2 and less than 5

[1] FALSE FALSE TRUE TRUE FALSE FALSE


> x[(x > 2) & (x < 5)] # Finds which values
                         are greater than 2 and
                         smaller than 5
[1] 3 4
```

So, now, I hope I have made this aspect clear and I have shown you that how you can use this and after this, I try to take here one more example and try to show you that how the things are working and through this example, I will try to give you some more operations, which can be done on this R software with the logical operators.

So, I try to consider here a data vector 1 to 6. So, I am writing here 1 colon 6. So, this is going to give us the values like 1, 2, 3, 4, 5, 6 it is something like I can also write x as data vector x equal to c parenthesis 1, 2, 3, 4, 5 and 6. Now after this I have a condition the condition here is that x greater than 2 and x smaller than 5. So, first you have to see what you want. You essentially want that this condition should be checked over each and every value in this data vector x.

So, for that now you have to use here the single & operator. So, this condition is going to check whether the values are greater than 2 and less than 5 for each of the element in the data vector x. So, now, you know that what will happen it will try to choose here the value here x and x, 1 is greater than 2 and 1 is less than 5 the answer will be here FALSE now you can verify it very easily, then it will try to pick up the second value it will try to see whether 2 is greater than 2 and 2 is smaller than 5 this value will also come out to be here FALSE.

Then it will try to take here x equal to 3 and it will try to check here the condition that whether 3 is greater than 2 and 3 is smaller than 5 and this condition will come out to be here TRUE. So, now, you can see here this answer is coming here like this. So, these are here this is corresponding to x equal to 1, the second value is corresponding to x equal to 3, third value is corresponding to x equal to 3 and fourth value is corresponding to x equal to 4, fifth value is corresponding to x equal to 5 and the sixth value is corresponding to x equal to 6.

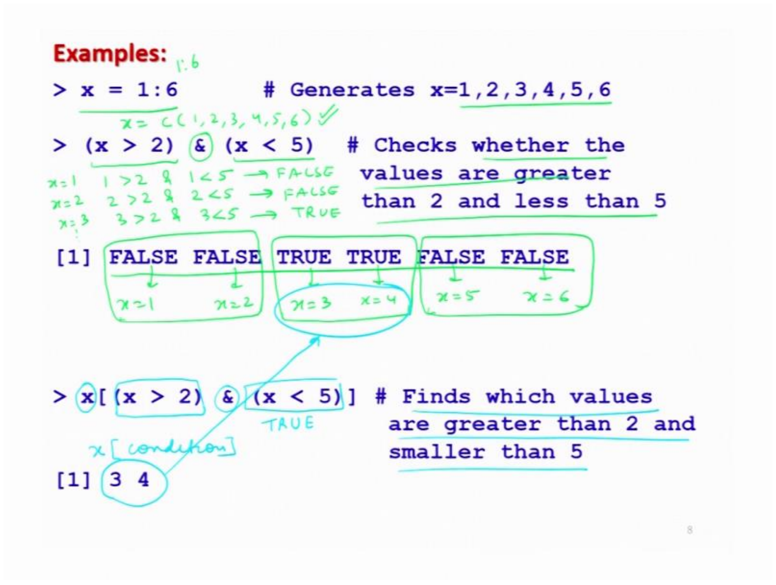
So, now you can see here that it is indicating that for the values of x, 1 and 2 this condition is FALSE, for the value 3 and 4 this condition is TRUE and for value 5 and 6 once again the condition is FALSE that we can see for x 1 and 2, for x 3 and 4 and for x 5 and 6. So, that is how we can check a condition over a data vector.

Now after this suppose you want to find that, which are the values in this outcome for which the condition is TRUE, right. So, you can see here these are there are two values here 3 and 4 for which the value is TRUE. So, now, you want to find these values in the R software in an automated way.

And this you can imagine well I am trying to take here only the six values. So, I can show you all this logic and manipulations manually, but suppose you are working with thousands value million values billion values in a data file then how can you check it manually that which are the values for which the condition is TRUE, right. So, in that case you need to work only through the programming and the programming is going to help you in finding out the values in the outcome, which are showing the outcome as TRUE.

So, in order to do it now I am going to explain you how you can do it. So, suppose I want to know what are the values in x for which the condition x greater than 2 and x less than 5 is TRUE, right ok. I repeat it once again because for the sake of understanding try to understand and try to look at my pen, I consider the data vector here x and I want to find out what are the values in this x for which the condition x greater than 2 and the condition x smaller than 5 is TRUE.

(Refer Slide Time: 37:14)



Examples:

```
> x = 1:6          # Generates x=1,2,3,4,5,6
> (x > 2) & (x < 5)    # Checks whether the
                         values are greater
                         than 2 and less than 5
[1] FALSE FALSE TRUE TRUE FALSE FALSE

> x[(x > 2) & (x < 5)] # Finds which values
                         are greater than 2 and
                         smaller than 5
[1] 3 4
```

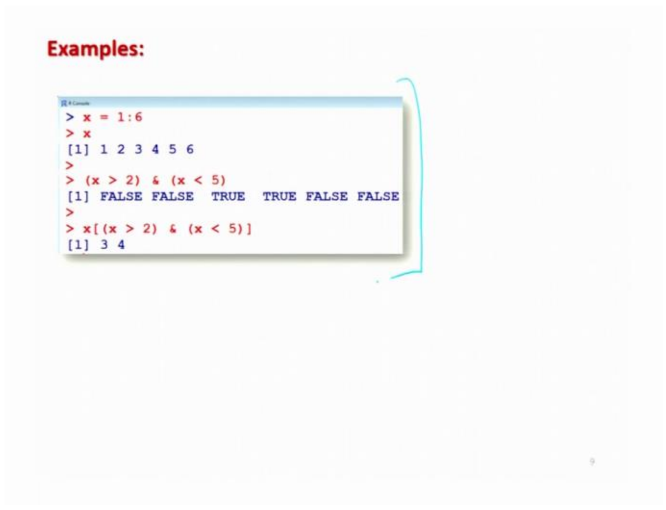
1:6
x= c(1,2,3,4,5,6) ✓
x=1 1>2 & 1<5 → FALSE
x=2 2>2 & 2<5 → FALSE
x=3 3>2 & 3<5 → TRUE
x=1
x=2
x=3
x=4
x=5
x=6
TRUE
x[condition]
8

So, if you try to see how I have expressed it. First I have expressed here the data vector in which I want this condition to be found and then I am trying to write down here a square bracket and under and inside this square bracket I try to write down here the condition. Well, this is the fundamental either your condition is related to & operator or say or operator that will remain the same this methodology will remain the same.

So, now if you try to do it and if you try to execute it on the R console you get here the values 3 and 4. So, you can see here that these are the two values, which you had found manually right. So, now, the rule is very simple in case if you want to judge whether a condition is TRUE or FALSE, you can simply write the condition and if you want to know that how many values in that data vector for which you have found the condition to be TRUE you have to write in this particular way.

Just write the value of the data vector or the symbol of the data vector and inside the square brackets try to write down the condition and this will give you the values for example, here this finds out that which values are greater than 2 and smaller than 5, right. So, this is how it actually works.

(Refer Slide Time: 39:02)

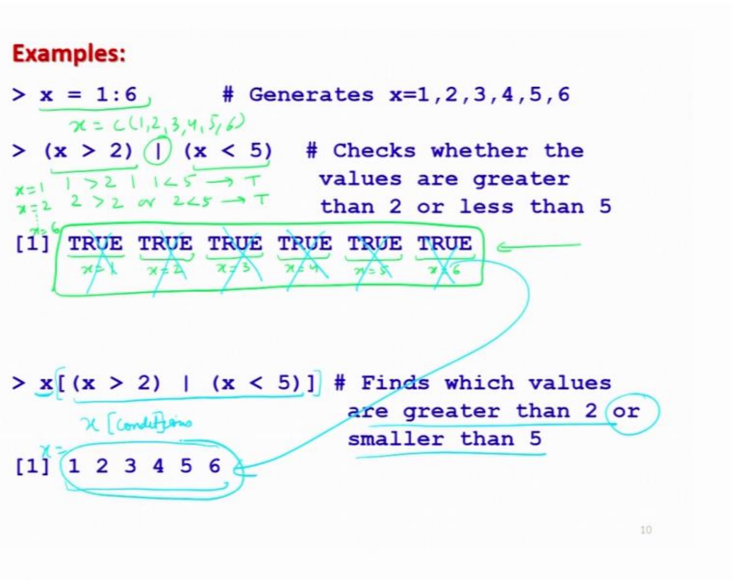


Examples:

```
> x = 1:6
> x
[1] 1 2 3 4 5 6
>
> (x > 2) & (x < 5)
[1] FALSE FALSE  TRUE  TRUE FALSE FALSE
>
> x[(x > 2) & (x < 5)]
[1] 3 4
```

And this is here the screenshot, right.

(Refer Slide Time: 39:09)



Examples:

```
> x = 1:6          # Generates x=1,2,3,4,5,6

> (x > 2) | (x < 5)   # Checks whether the
                        values are greater
                        than 2 or less than 5
[1] TRUE TRUE TRUE TRUE TRUE TRUE



> x[(x > 2) | (x < 5)] # Finds which values
                         are greater than 2 or
                         smaller than 5
[1] 1 2 3 4 5 6
```

x = c(1,2,3,4,5,6)
x=1 1 > 2 | 1 < 5 → T
x=2 2 > 2 or 2 < 5 → T
x=6
x=1
x=2
x=3
x=4
x=5
x=6
x [conditions]
x =

16

Now let me first repeat this example with the R operator and after that I will try to show it on the R console also. Well, since these screenshots are there. So, it should not be very difficult for you to understand.

So, now, I once again take the same data vector here x equal to 1 to 6 and I try to take here a condition that x greater than 2 or x is smaller than 5 and I want that in this data vector x is equal to like 1, 2, 3, 4, 5 and 6. I want R to check this condition that x greater than 2 or x is smaller than 5 is TRUE or FALSE. So, what will happen? The same operation at x will try to choose the first value 1 and it will try to check here 1 is greater than 2 or 1 is less than 5 and since you have given here only the single operator. So, now, the control will go to the next value also and it will try to see here for what happened for x equal to 2 and it will try to check here whether 2 is greater than 2 or 2 is greater than FALSE.

And in this case you can find out the answer will come out to be here TRUE in the first case and TRUE in the second case and similarly you can continue with x equal to 6 and you will find the answers and this outcome is given here. So, you can see here this value is for x equal to 1 the second value is TRUE, which is indicating the outcome for the x equal to TRUE 2 and the third value is here TRUE, which is indicating the outcome of the value x where x is equal to 3.

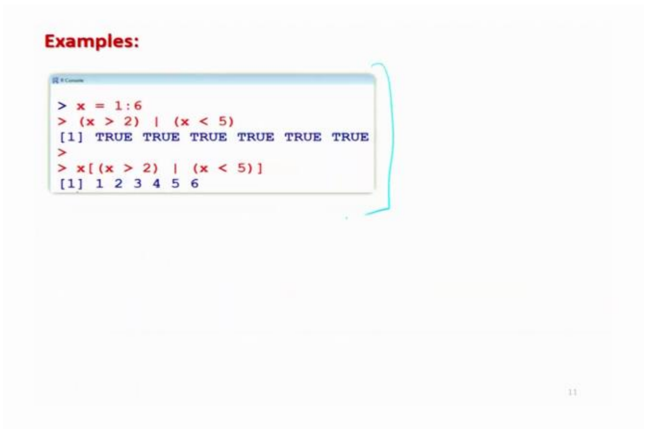
And similarly the fourth value is indicating the outcome for x equal to 4 which is TRUE the fifth value is indicating the outcome for x equal to 5 which is the fifth value in the data vector and then finally, this last TRUE it is indicating the value of x for x equal to 6. So, this TRUE for is the answer when x equal to 6, right. So, this is how it will actually work. Now in case if I want to find out that what are the values in this outcome for which the answer is TRUE.

So, you can see here, this condition is TRUE for all the values of x equal to 1, 2, 3, 4, 5 and 6, but if I want to do this operation on the R software using the R command, then I have to follow the same rule, same methodology, which I just explained you in the case of and operator and you simply try to write down here the condition whatever whatsoever is the condition and then you write down here the data vector and then close it inside the square bracket.

So, the rule is very simple try to write down the data vector and inside the square bracket try to write down the conditions and you can see here because this TRUE is coming for x equal to 1, x equal to 2, x equal to 3, 4, 5 and 6. So, the same thing is given here these are the values of x for which the condition is TRUE, which condition? That the values are greater than 2 and or they are smaller than 5.

So, all those values which are greater than 2 or smaller than 5 they are mentioned here in this outcome, ok.

(Refer Slide Time: 43:11)

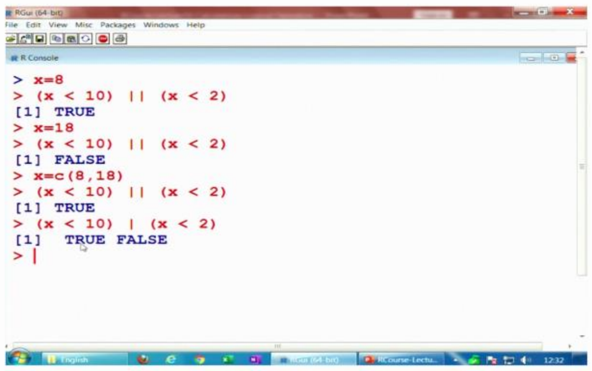


Examples:

```
> x = 1:6
> (x > 2) | (x < 5)
[1] TRUE TRUE TRUE TRUE TRUE TRUE
>
> x[(x > 2) | (x < 5)]
[1] 1 2 3 4 5 6
```

Now you can see that this is not a very difficult operation and if you try to do it on the R console this type of is outcome will be there, but now before going further let me try to show you these operations in the R console also. So, if I try to copy this conditions and then I will try to operate it on the R console.

(Refer Slide Time: 43:35)

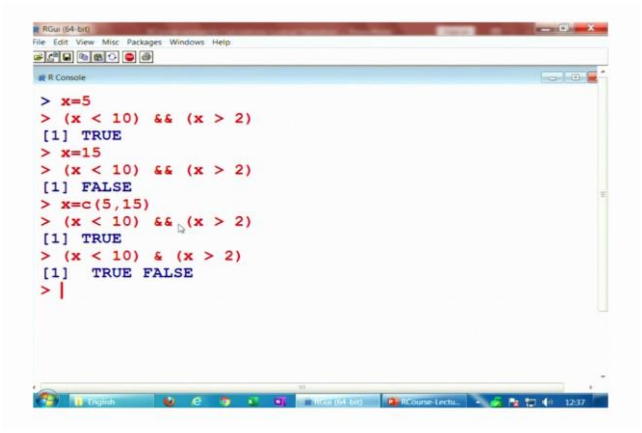


```
> x=8
> (x < 10) || (x < 2)
[1] TRUE
> x=18
> (x < 10) || (x < 2)
[1] FALSE
> x=c(8,18)
> (x < 10) || (x < 2)
[1] TRUE
> (x < 10) | (x < 2)
[1]  TRUE FALSE
> |
```

So, if I try to write down here x equal to here 8 and then I try to see here what is this condition this is here TRUE, and now in case if you want to choose here x equal to 18 then after that you have to type the condition once again. So, I type the condition this is here FALSE and now in case if I try to define here x as here the data vector 8 and here 18 and then I try to give this condition you can see here this is only TRUE because there are here two vertical lines, two operators.

But if I try to change the same condition with single operator like this then if you try to enter you can see here this is giving you the answer TRUE and FALSE.

(Refer Slide Time: 44:33)



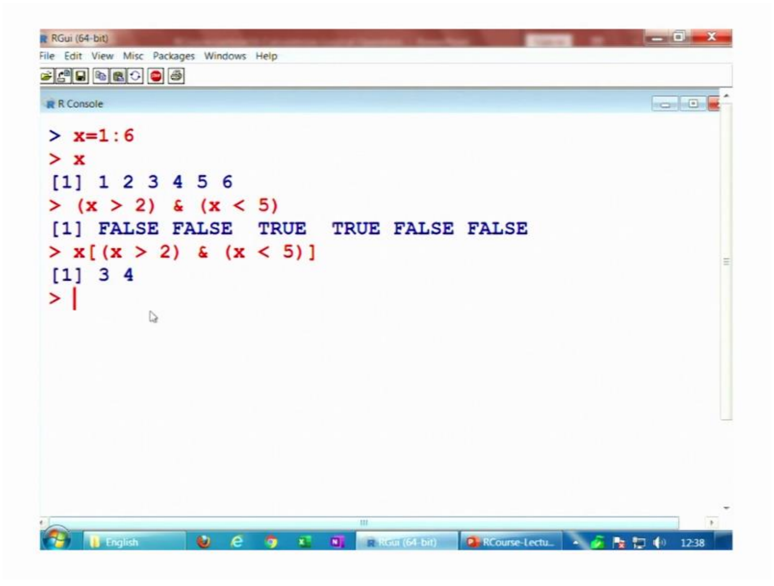
```
> x=5
> (x < 10) && (x > 2)
[1] TRUE
> x=15
> (x < 10) && (x > 2)
[1] FALSE
> x=c(5,15)
> (x < 10) && (x > 2)
[1] TRUE
> (x < 10) & (x > 2)
[1]  TRUE FALSE
> 
```

So, now, if I try to take here other condition and I have choose here x equal to 5 and I try to operate it with the & operators you can see here when I am using two & operator it is giving value TRUE and when I try to choose here the value x equal to 15 then this condition is giving me an answer FALSE.

And when I try to choose both the values inside a data vector here like this 5 comma 15, then you can see here that this is giving me here the values, which is here only single value because I have used here two & operators, but if I try to use here only one & operator it will give me here the answer TRUE and FALSE, right.

So, ok after this I try to take here this example where I am trying to use six numbers.

(Refer Slide Time: 45:27)

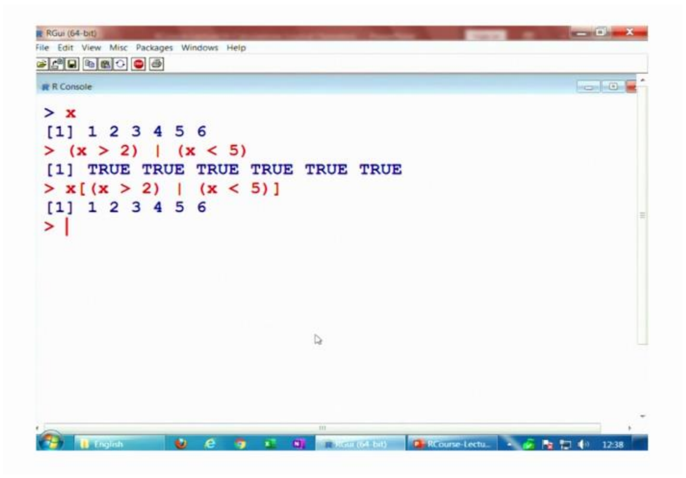


```
> x=1:6
> x
[1] 1 2 3 4 5 6
> (x > 2) & (x < 5)
[1] FALSE FALSE  TRUE  TRUE FALSE FALSE
> x[(x > 2) & (x < 5)]
[1] 3 4
> |
```

So, x is going to be here 1 colon 6 you can see here. So, now, value is here x 1, 2, 3, 4, 5, 6 like this and if I try to execute this single and operator this is coming out to be FALSE FALSE TRUE TRUE FALSE FALSE and if I try to and suppose if I want to know that which are the values here in this outcome for which the values are TRUE then I simply have to write here x and inside the square bracket I have to write down the condition and you can see here this is coming out to be here 3, 4.

And similarly in case if I want to find out this condition for the or operator here, you can see here that this is here like this.

(Refer Slide Time: 46:14)

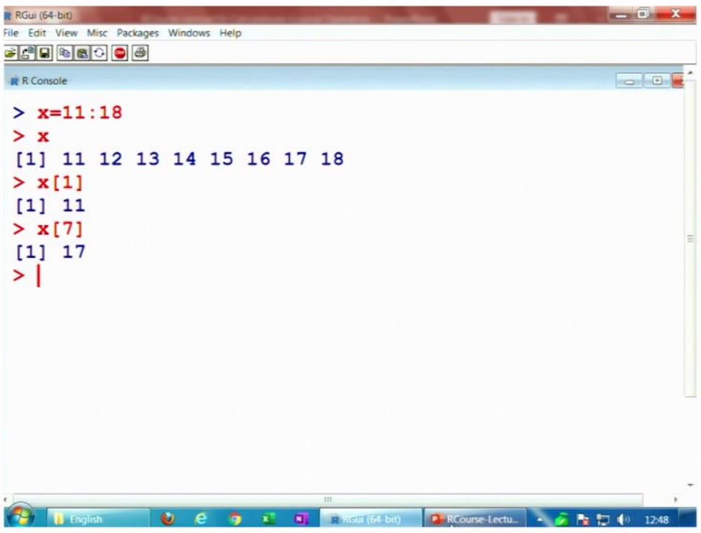


```
> x
[1] 1 2 3 4 5 6
> (x > 2) | (x < 5)
[1] TRUE TRUE TRUE TRUE TRUE TRUE
> x[(x > 2) | (x < 5)]
[1] 1 2 3 4 5 6
> |
```

That I try to take here x here as say 1 to 6 and if I try to operate this condition then it is operated over the entire data vector and all the condition are coming out to be TRUE and if I want that which of the values are here TRUE.

Then I have to simply write down the data vector x and inside the square bracket I have to write down the condition and it will give you the answer 1, 2, 3, 4, 5, 6.

(Refer Slide Time: 46:39)

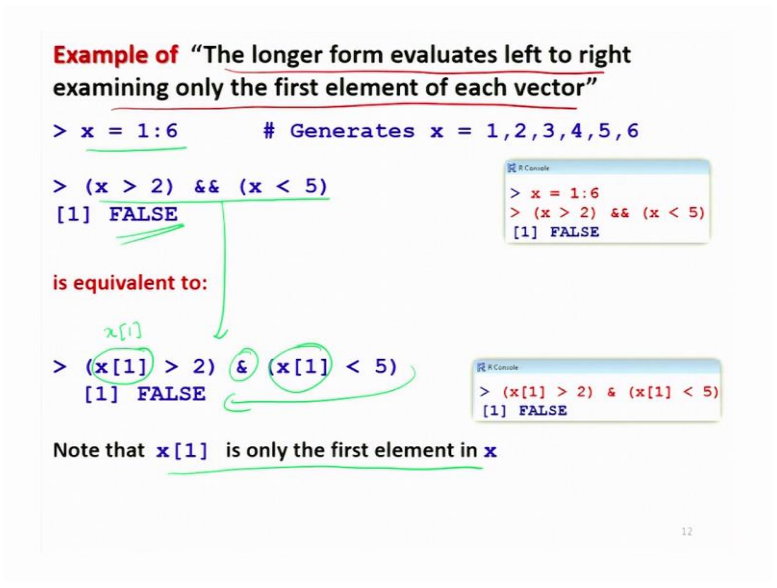


```
> x=11:18
> x
[1] 11 12 13 14 15 16 17 18
> x[1]
[1] 11
> x[7]
[1] 17
> |
```

And now before I move further let me try to give you here one more concept so, that I can explain you something if you try to see I am taking here value of x from 11 to 18 like this, now if you try to write down here x and inside the square bracket if you write here 1 then you see what you are getting.

You are getting the value here 11; that means, this one inside the square bracket is the location in the data vector and this expression is giving you the first value in the data vector. And similarly if you try to write down here x inside the bracket if you write here 7, then you will get here the seventh value in that data vector.

(Refer Slide Time: 47:20)



Example of "The longer form evaluates left to right examining only the first element of each vector"

```
> x = 1:6          # Generates x = 1,2,3,4,5,6

> (x > 2) && (x < 5)
[1] FALSE
```

is equivalent to:

```
> (x[1] > 2) & (x[1] < 5)
  [1] FALSE
```

Note that `x[1]` is only the first element in `x`

```
> x = 1:6
> (x > 2) && (x < 5)
[1] FALSE
```

```
> (x[1] > 2) & (x[1] < 5)
[1] FALSE
```

x[1]
R Console
R Console
12

So, if you have understood this concept then I would try to give you here one example to explain that what is the meaning of this longer form evaluates left to right examining only the first element of each vector which I promised you in the beginning of the lecture.

So, now you have taken x from 1 to 6 now if you try to take a x greater than 2 and with double & operator it is x smaller than 5, then it is giving you an answer FALSE. So, you know that why this is happening, but what is happening that I want to explain you. If you simply try to take the first element of the data vector, which is indicated by here x and inside the parenthesis brackets 1 and you try to execute this condition. This condition here with only with the first element here like this using the single & operator you get the same outcome.

So, this x 1 is indicating only the first element. So, this is being operated only on the first element, right. So, anyway I have explained you this concept with couple of example and I hope now you are clear that how logical operators are going to work in the R software and they are very useful actually you will see when you are trying to do the programming many at many many places you will try to use them.

And, but now your job is that after I finish this lecture try to take some values try to understand what is the difference between mathematical and logical operators, how do they operate and how do they give the outcome, what is their evaluation process and try to do it manually and then try to do it inside the R software also because finally, you will not have an opportunity to look at them manually. So, this practice will give you confidence that whatever you are thinking R is doing the same thing. So, you try to practice it and I will see you in the next lecture. Till then, goodbye.