

**Foundations of R Software**  
**Prof. Shalabh**  
**Department of Mathematics and Statistics**  
**Indian Institute of Technology, Kanpur**

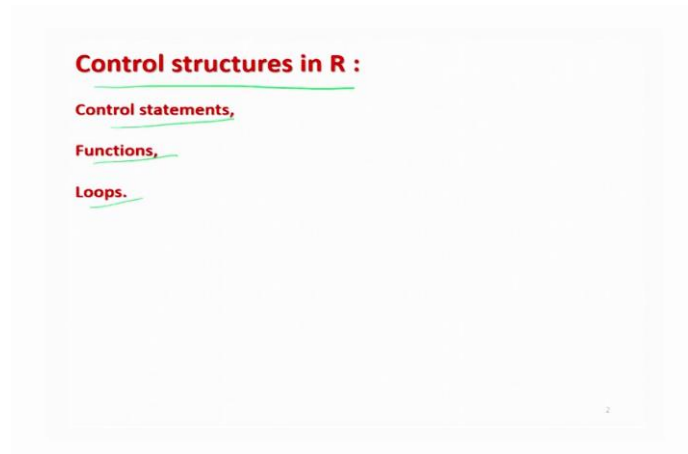
**Basics of Calculations**  
**Lecture - 20**  
**Conditional Executions - Nested If else If and If-Else**

Hello friend, welcome to the course Foundations of R Software and you can recall that in the last lecture we started our discussions on the Conditional Execution and we had considered two commands If and If Else. And you had understood that under what type of conditions we use such conditional executions. So, now in case if you try to recall how we proceeded; first we considered the condition if, where you have to execute it only when the condition is true.

Then you considered the condition if else, where you would like to proceed in two ways; basically if the condition is true, then do like this and if the condition is false, do like this. But now suppose you have more than two options; if you have two options, you can control it by say true or false. But in case if you in case if you have more than two options and if you have more than two conditions and based on that, you have to execute something, then how are you going to do?

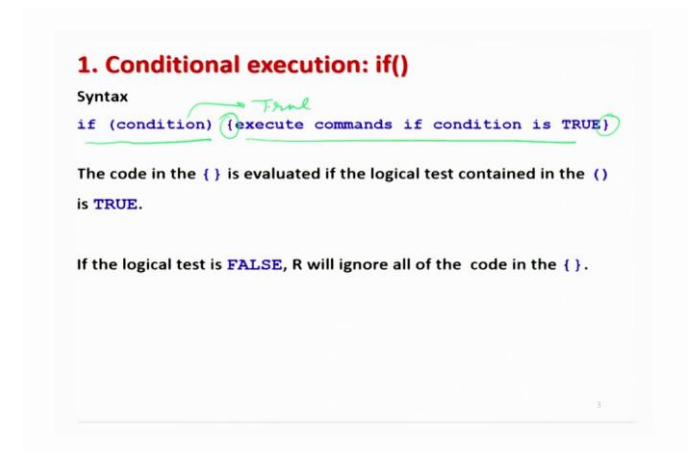
So, one simple concepts that comes to our mind automatically is that can we extend the if else condition? That you have used if, then else. Now, can you add more condition if else, if else and so on? Yes, so we are going to talk about such if else condition which repeat and then I will tell you about one more conditional execution, in which the executions are happening if the condition is true or false; that is also similar to if else condition, but it is syntax is different.

(Refer Slide Time: 02:26)



So, we begin this lecture and I try to explain you these two new concepts. As you had initiate in the initiated in the last lecture that we are now trying to discuss the control structures, in which we would like to go for control statement, functions and loops and these control structures are needed when you are trying to do the programming.

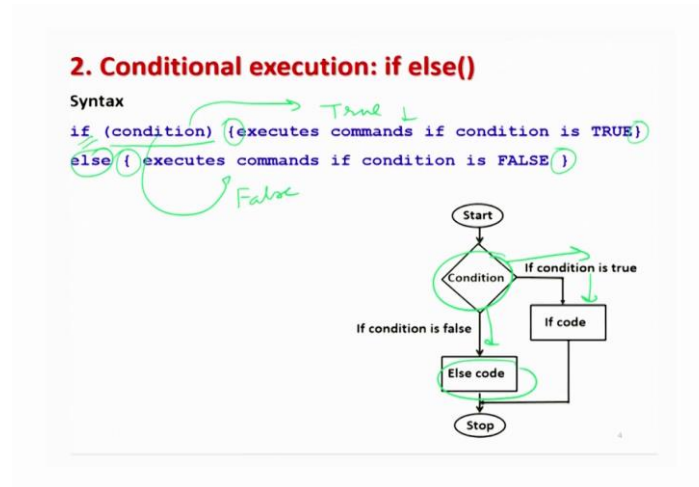
(Refer Slide Time: 02:46)



So, at this moment we are considering the conditional executions and just for your recollection, I will try to give you here a quick review of the commands.

So, in the last lecture, we had learnt about the if condition and the rule was you have to write the condition like this if and then within the parenthesis you have to write the condition. In case if the condition is true, then whatever you are trying to write inside this curly brackets that will be executed and if false, then after this the program will stop and there would not be any outcome.

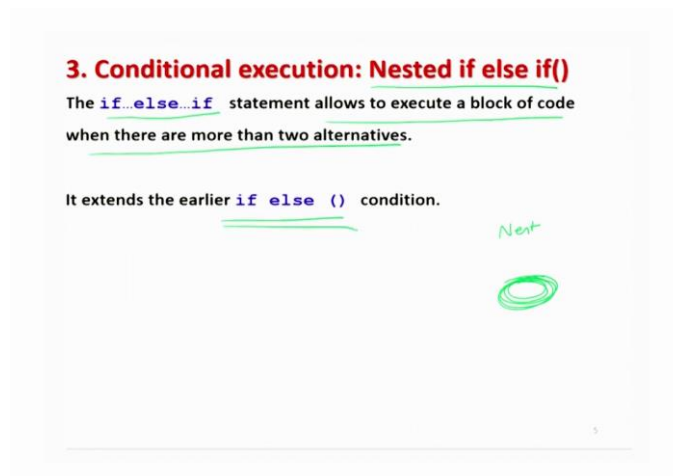
(Refer Slide Time: 03:17)



And after that we had considered the conditional execution, in which we were trying to say that in case if the condition is true; then some codes have to be executed and if the condition is false, then some other codes have to be evaluated.

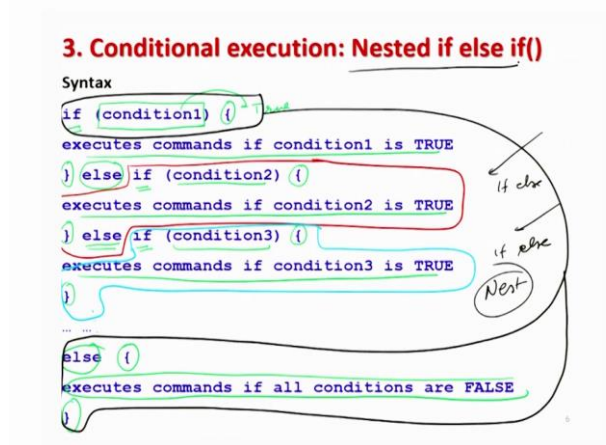
So, the syntax was you try to write down here if and then within the parenthesis, you try to write down the condition; in case if the condition is true, then whatever you write just after this inside the curly brackets that is going to be executed. And if this condition is false, so then you try to write down here else and then write down the condition inside the bracket, which are going to be evaluated in case if the condition comes out to be false and this is how this if else condition was working.

(Refer Slide Time: 04:08)



Now, I am going to work on another aspect, which is a sort of extension of the if else condition and this is called nested if else. So, do you know what is the nest and have you ever observed that how a bird makes the nest? It will try to take some sticks and then try to put one stick over each other and so on, right. So, in this case we try to use the if else if etcetera couple of time and this statement allows us to execute a block of code when there are more than two alternatives, right.

(Refer Slide Time: 04:53)



So, as I said this is a sort of extension of the if else condition. So, now, how you have to give this condition; suppose you have got couple of conditions say condition 1, condition 2, condition 3 and so on. And based on the condition; whether the condition is true or false, you have to execute certain commands. So, in case if you try to see, the way I am trying to write down this syntax if you try to understand; then it will be very easy for you to implement it. So, I try to use the same concept, which I used in the case of if else.

So, in the case of if else what happens that, I will try to write down here if and then in the parenthesis I will try to write down the condition and after this, whatever has to be executed, if this condition is true is to be written. So, suppose my first condition I am trying to indicate condition 1. So, now, if this condition is true, then I try to write whatever is to be executed inside the curly bracket. So, I write here executes commands if condition 1 is TRUE. And in case if the condition is false, then it moves further and you write here else; just like as you have done in the earlier case.

But earlier what you did, after the else you just wrote the condition inside the curly bracket; but now you try to continue with the, if else condition. So, I try to do here say else, after this else I try to repeat and I try to write down here if; the second condition say for example, I write condition 2 inside the parenthesis and then whatever is to be executed, I am trying to write down here inside the curly brackets.

So, this is the statement like executes command if condition true, 2 is TRUE. And suppose if this is also not true, then again I have to just repeat it; I will simply try to write down here else and then if the condition 3 and then after this whatever I want to execute, I have to write down inside the curly bracket like execute command if condition 3 is TRUE. And this will continue and finally, you have to write down here else and after that you have to write down within the curly brackets whatever is to be executed, if all the earlier conditions like condition 1, condition 2 etc. they were not true, right.

So, whatever you want to execute finally that is written here that, execute command if all conditions are false. So, in case if you try to see the structure here carefully what happens? If you try to see here, you are trying to create here a structure like as here if and then here else like this and then once again you are trying to create here a structure like here if and then here else and so on.

And finally, whatever you have means, you are essentially trying to start here if condition and you are trying to close it here. So, within these if you try to see here there are condition if else, if else etc. So, that is why this looks just like a nest and this type of execution is called as nested if else condition. So, I hope this is clear.

(Refer Slide Time: 08:17)

**3. Conditional execution: Nested if else if()**

**Example 1:**

```

x = 5
if ( x == 3 ) {
  x = x - 1;
} else if ( x < 3 ) {
  x = x + 5;
} else {
  x = 2 * x;
}
x
[1] 10
  
```

**Interpretation:**

- If  $x = 3$ , then execute  $x = x - 1$ .
- If  $x < 3$ , then execute  $x = x + 5$ .
- If  $x > 3$ , then execute  $x = 2 * x$ .

In this case,  $x = 5$ , so  $x > 3$ . Thus  $x = 2 * 5$

And now, let me try to take here some example by which I can explain you that, it is not a difficult thing to execute this condition and to get an outcome. Suppose I try to write here a function  $f(x)$  and this function takes value here  $x$  minus 1, if  $x$  is equal to 3 and it takes value  $x$  plus 5, if  $x$  is less than 3. And if  $x$  is not less than 3,  $x$  is not equal to 3, but  $x$  is greater than 3; that means otherwise, if these two conditions are not true, then it will simply find twice of  $x$ .

So, this type of function you have learnt and you are pretty comfortable. So, that is what I said in the beginning that when there are more than two condition, which are to be satisfied in terms of true and false, then we use this type of conditional execution.

So, I try to write down here this thing. Now, if you have understood and if you have practiced the if else condition, then writing the syntax for such a function is not difficult at all. So, what I have to do here that, I try to write down here if and then I try to write down here  $x$  equal to 3. And after this my aim is that, if this condition is true; then I want to replace  $x$  by  $x$  minus 1 and I try to write it down under inside the curly brackets.

Now, after this in case if this condition is false, then what will happen? If this condition here is false, then it will try to check else, if  $x$  is less than 3. And then whatever I want to execute, I try to give it inside the curly brackets and this is replace  $x$  is equal to  $x$  plus 5; that means you try to replace the value of  $x$  by the value  $x$  plus 5 and after this.

Now, there are two options, whether the condition this  $x$  less than 3; if this is true, then you are trying to execute  $x$  equal to  $x$  plus 5. But in case if this condition is here false; that means the first condition I am marking here this is false, the second condition this is here false. So, now, whatever you want to do, just try to write down here else and inside the curly bracket you try to write what you want to execute, that is  $x$  is equal to  $2x$ .

So, in case if you try to look here, I will try to use here black pen and then I will try to mark here and to show you the correspondence between the function and the program. So, if your condition was like this  $x$  minus 1 is equal to  $x$  is equal to  $x$  minus 1, if  $x$  equal to 3; so this condition has been executed here. And similarly, in case if you want to see what happens with  $x$  equal to  $x$  plus 5, that is executed when  $x$  is smaller than 3; so this is executed here, you can see here and this is here like this.

And after this you have here  $2x$ . So, then you are trying to write down here this  $2x$  as  $2x$  and this is here executed here. So, you can see here this is what we wanted to do that, if  $x$  is equal to 3, then execute  $x$  equal to  $x$  minus 1; if  $x$  is smaller than 3, then execute  $x$  is replaced by  $x$  plus 5 and in case if  $x$  is greater than 3, try to execute  $x$  is equal to  $2x$ . So, now I try to show you one example. I try to take here a value here  $x$  equal to 5, you can see here. And now, if you try to see here, where this  $x$  equal to 5 falls? 5 is greater than 3, this is the only condition which is here true.

So, what will happen, the control will come here first to 5 is exactly equal to 3, this comes out to be here false; then it comes to 5 is less than 3, this also comes out to be here false and then finally, it come to the third condition that else whatever is there, you just try to print here 2 into 5 and this answer here comes out to be here 10, which is obtained here, right. So, this is how this conditional execution works.

(Refer Slide Time: 12:46)

**3. Conditional execution: Nested if else if()**

**Example 2:**

```

x = 2
if ( x==3 ) {
x = x-1
} else if ( x < 3 ) {
x = x+5
} else { x = 2*x }
x
[1] 7

```

*Handwritten annotations:*  $2=3$  is false (F),  $2 < 3$  is true (T),  $2+5=7$ .

```

> x = 2
> if ( x==3 ) {
+ x = x-1
+ } else if ( x < 3 ) {
+ x = x+5
+ } else { x = 2*x }
> x
[1] 7
>

```

**Interpretation:**

- If  $x = 3$ , then execute  $x = x - 1$ .
- If  $x < 3$ , then execute  $x = x + 5$ .
- If  $x > 3$ , then execute  $x = 2 * x$ .

In this case,  $x = 2$ , so  $x < 3$ . Thus  $x = 2+5$

Now, so now, I try to take here one more value in the same example. So, I try to take here  $x$  equal to 2. So, now, you can see what will happen; the condition will come here 2 is equal to 3, this is false. So, it will not execute here this thing, but it will come to here second part. Now, it will try to check here 2 less than 3, the answer comes out to be here true.

So true, so whatever is written here inside this curly brackets, this is going to be executed and you will get here the value here 2 plus 5 equal to 7. And after this the control will not go to the third condition, because this condition is true and as soon as the condition

become true, the execution is done. And you can see here the outcome comes out to be here 7 and this is exactly what I explain you.

(Refer Slide Time: 13:48)

### 3. Conditional execution: Nested if else if()

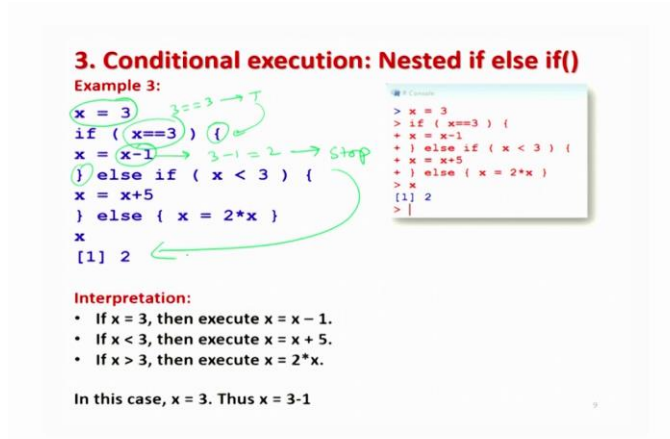
**Example 3:**

```
x = 3
if ( x==3 ) {
  x = x-1
} else if ( x < 3 ) {
  x = x+5
} else { x = 2*x }
x
[1] 2
```

**Interpretation:**

- If  $x = 3$ , then execute  $x = x - 1$ .
- If  $x < 3$ , then execute  $x = x + 5$ .
- If  $x > 3$ , then execute  $x = 2*x$ .

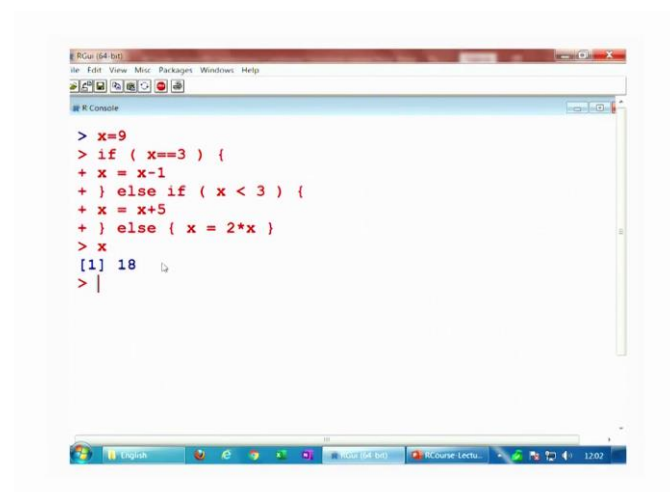
In this case,  $x = 3$ . Thus  $x = 3 - 1$



And similarly if you try to take here one more value here  $x$  equal to 3, then what will happen? The control will start execution and control will come here at the first place; now it says here 3 is equal to 3, answer comes out to be here true.

So, as soon as it is true, whatever is written in the first curly bracket that is 3 is going to be replaced by 3 minus 1 equal to 2 that will be executed and after this the program will stop and you can see here this is the outcome which is coming here. So, now, let me try to show you this example on the R console, so that you become more confident and then I will try to move further.

(Refer Slide Time: 14:40)



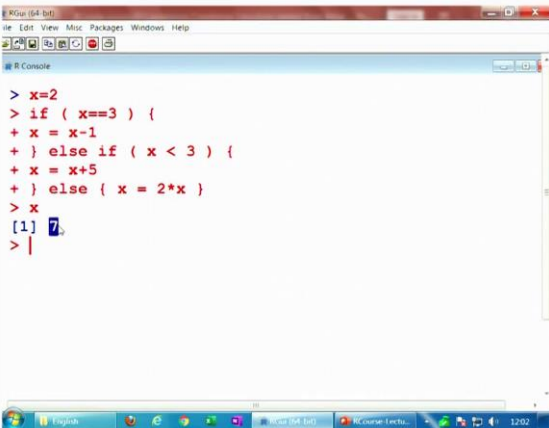
```
> x=9
> if ( x==3 ) {
+ x = x-1
+ } else if ( x < 3 ) {
+ x = x+5
+ } else { x = 2*x }
> x
[1] 18
> |
```



So, I try to copy this command here, so that I can save some time. And you can see here if I try to write down here x equal to suppose 9 and if I try to execute this thing and I try to press here x, the value comes out to be here 18.

Why? Because if you try to see the x is replaced by here 2 into x, that is 2 into 9 which is 18.

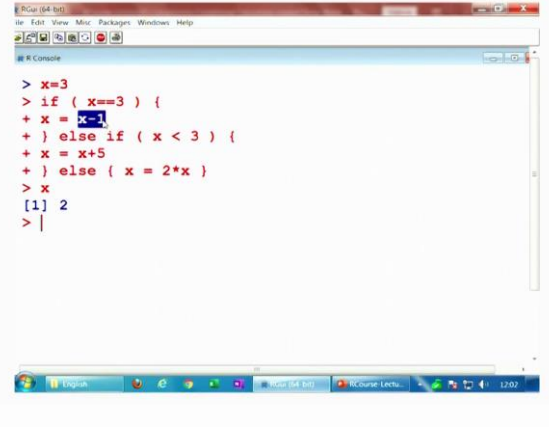
(Refer Slide Time: 15:02)



```
RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
> x=2
> if ( x==3 ) {
+ x = x-1
+ } else if ( x < 3 ) {
+ x = x+5
+ } else { x = 2*x }
> x
[1] 7
> |
```

Now, similarly if I try to take here suppose x equal to here say 2 and if I try to repeat it; then try to see what is the value of here, x this comes out to be 7 y, because x is equal to 2, which is smaller than 3. So, the second condition is satisfied and it becomes a 2 plus 5, so this is here 7.

(Refer Slide Time: 15:24)



```
RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
> x=3
> if ( x==3 ) {
+ x = x-1
+ } else if ( x < 3 ) {
+ x = x+5
+ } else { x = 2*x }
> x
[1] 2
> |
```

And similarly, if you try to take here x exactly equal to 3 and you try to execute the same command; then the value of x comes out to be here 2. Why? Because at x equal to 3, this condition become true and this x minus 1 is executed, which gives you the value 3 minus 1 equal to 2.

(Refer Slide Time: 15:50)

**4. Conditional execution: ifelse()**

Syntax

```
ifelse(test, yes, no)
```

Condition

if else

- Vector-valued evaluation of conditions .
- For the components in the vector-valued logical expression **test** which provide the value **TRUE**, the operations given by **yes** are executed.
- For the components in the vector-valued logical expression **test** which provide the value **FALSE**, the operations given by **no** are executed.

10

So, you can see here it is not difficult at all to work with this nested if else condition. And now I will give you one more option, where you can do the conditional execution. And this is also called as if else condition, but there is a difference; if you try to see earlier, we had written if and then else and there was a blank space between the two. But now there is no blank space, I am writing i f e l s e without any blank space, all in lower case alphabets.

So, this execution works actually just likes if else condition; but there is a minor difference, the difference is like this. Then inside the parenthesis you want to execute, whatever you want to execute that has to be specified. And the way it is specified is like this that, you try to give here a condition and this condition is to be tested.

In case if the condition is found to be true; that means the answer is yes, then whatever condition or whatever commands you are given here under the yes, they will be executed. And if this condition results in a false, then whatever you have written as here no; the syntax and commands whatever you write here, they are going to be executed.

So, that is a very simple statement and it is just like an if else statement that you did earlier; but the only difference is that here you are trying to write if else together and then within one parenthesis, you are trying to first write the condition and if the condition is true or false, based on that you are trying to specify the commands which are separated by comma, right.

So, this is actually useful when we are trying to make vector value evaluations of conditions. If you try to see in the earlier conditions I had always told you that they are useful when you are trying to deal with the scalar value; but in this case, you can handle the vector value.

So, now, the first concept is clear that, under what type of condition you are going to use if else condition and this type of if else condition; yeah just be careful because I am always using the word if else and I cannot pronounce if else something like if else yeah that, will not look that will not sound actually good.

So, anyway, so this is the first advantage of this condition and the way it is happening as I said that, the components in the vector value logical expression, which is given under the test; they will when they provide the outcome to be true, then whatever is the commands given under the yes, they are executed. And if they provide the answer to be false; then the commands which are under the no, they will be executed.

(Refer Slide Time: 19:03)

**4. Conditional execution: ifelse()**

**Example 4:**

```
> x = 1:10
> x
[1] 1 2 3 4 5 6 7 8 9 10
> ifelse(x < 6, x^2, x+1)
[1] 1 4 9 16 25 7 8 9 10 11
```

**Interpretation**

- If  $x < 6$  (TRUE), then  $x = x^2$  (YES).
- If  $x \geq 6$  (FALSE), then  $x = x + 1$  (NO).
- So for  $x = 1, 2, 3, 4, 5$ , we get  $x = x^2 = 1, 4, 9, 16, 25$
- For  $x = 6, 7, 8, 9, 10$ , we get  $x = x + 1 = 7, 8, 9, 10, 11$

*Handwritten notes on the slide:*

$f(x) = \begin{cases} x^2 & x < 6 \\ x+1 & \text{otherwise} \end{cases}$

$x = 1, 2, \dots, 10$

$x = 1, 1 < 6 \rightarrow T \rightarrow 1^2 = 1$

$x = 2, 2 < 6 \rightarrow T \rightarrow 2^2 = 4$

$x = 7, 7 < 6 \rightarrow F \rightarrow 7 + 1 = 8$

$x = 8, 8 < 6 \rightarrow F \rightarrow 8 + 1 = 9$

$x = 10, 10 < 6 \rightarrow F \rightarrow 10 + 1 = 11$

→ Stop

So, let me try to take here some example and then try to show you that how this actually works and it will make the things very clear. So, let me try to take here set of number  $x$  equal to 1 to 10, so this is the number 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. And now what I want, it is something like this;  $f(x)$  is equal to  $x^2$  if  $x$  is equal to 1, 2 up to here 5 and it is going to be here  $x + 1$  if  $x$  is equal to 6, 7, 8, 9 and 10 like this. So, this is the function that we want to evaluate.

So, now I try to write down this condition here as say if else without any blank space; then within the parenthesis I try to write down first the condition, the condition here is  $x$  is less than 6. So, what we want here that if  $x$  is less than 6, then it is to be here like here  $x^2$  in general; well I have because I have taken here the values 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, so I have written here like this, but that can be valid or any real values also, right. And in case if not; means otherwise this is going to be here  $x + 1$ .

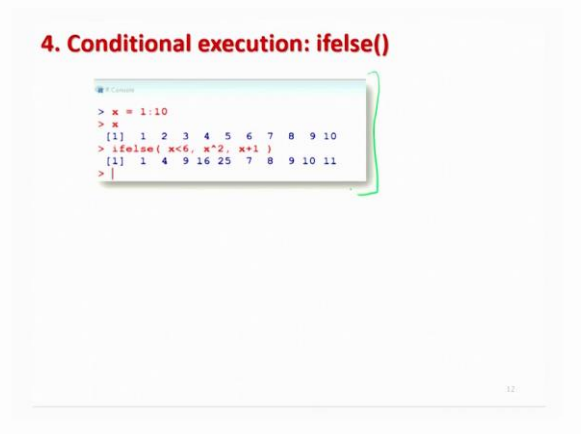
So, now this is the condition where  $x$  is less than 6. Now, this condition will have two outcomes, whether this condition is true; then the outcome is going to be  $x^2$  and if the condition is false, then the outcome is going to be  $x + 1$ . So, now, what will happen that, as soon as you execute it the control will first come to the input value, which is here  $x$  equal to 1, 2 up to here 10.

Now, it will first try to choose here the first value 1 and it will try to test whether 1 is less than 6 true or false; the answer here is true and when true, this will become here 1 square and the outcome will be here 1. Similarly, if we try to come to the next value  $x$  is equal to 2 and then it will try to test the condition whether 2 is smaller than 6 or not; the answer comes out to be here true and this will give us an outcome 2 square, which is here 4 and it will continue.

Then suppose it takes somewhere here the value here 7; so 7 is less than 6, answer is false. So, when it is false, then it will give you the outcome here as here say  $x + 1$ , right. So, this will become here 7 plus 1 equal to here 8. And similarly if you try to take here one more value here 8 and then it will say here 8 less than 6, this condition is false and then the outcome will become here 8 plus 1 is equal to 9 and this will come up to 10. And then 10 is less than 6, this is false; the outcome will become here 10 plus 1, which is 11 and after this it will stop, because there are no more values in the data vector  $x$ .

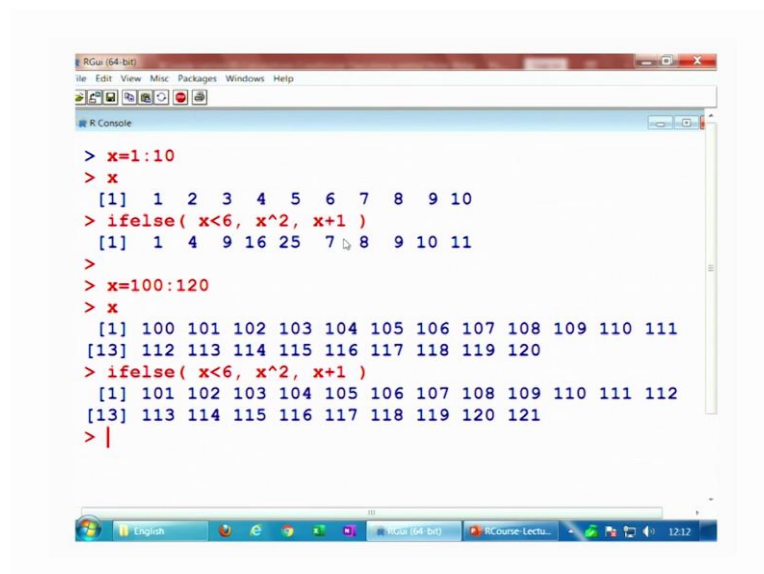
So, this is how it is going to work. So, you can see here if  $x$  is less than 6 that is true, then  $x$  equal to  $x$  square, which will be operated by the command under the yes. If  $x$  is greater than or equal to 6; that means the condition is false, then  $x$  will be replaced by  $x$  plus 1 and  $x$  equal to  $x$  plus 1 has to be given under the option no. So, you can see here for  $x$  equal to 1, 2, 3, 4, 5; we get here these value 1, 4, 9, 16, 25 and for  $x$  equal to 6, 7, 8, 9, 10, you get here the value 7, 8, 9, 10 and 11, right.

(Refer Slide Time: 23:16)



So, and the same outcome is here also, you can see here. Now, I try to show you this thing on the R console and you can see here this is the screenshot of the same outcome here. So, let me try to just copy this command, so that I can save some time.

(Refer Slide Time: 23:33)



And I try to come here on the R and I try to define here x is equal to 1 to 10. So, you can see here x is here like this. And if I try to execute this condition, this gives you here like this, right. And even if you try to change here; suppose if I try to define here say 100 to say 120, then x is here like this and if you try to consider here, you can see here this execution is obtained here. So, now, you can see that it is not a very difficult thing in R to execute such operations.

(Refer Slide Time: 24:06)

**4. Conditional execution: ifelse()**

**Example 5:**

```
x = c(7, 9, 8, 4)
ifelse(x %% 2 == 0, "even number", "odd number")
```

[1] "odd number" "odd number" "even number" "even number"

**Modulo Division**- Finds the remainder after division of one number by another.]

**Interpretation**

If the remainder of x divided by 2

- is 0, then print "even number" (YES) .
- is not equal to 0, then print "odd number" (NO).

So for x = 7, 9, we get x = "odd number" and  
for x=8, 4, we get x= "even number"

*Handwritten notes on the slide:*

- $x = 7, 7 \div 2, \text{rem} = 1 \Rightarrow 0 \Rightarrow \text{F} \rightarrow \text{odd}$
- $x = 9, 9 \div 2, \text{rem} = 1 \Rightarrow 0 \Rightarrow \text{F} \rightarrow \text{odd}$
- $x = 8, 8 \div 2, \text{rem} = 0 \Rightarrow 0 \Rightarrow \text{T} \rightarrow \text{Even}$
- $x = 4, 4 \div 2, \text{rem} = 0 \Rightarrow 0 \Rightarrow \text{T} \rightarrow \text{Even}$

And now let me give you one more example and after that I can explain you the utility of this if else condition. Suppose your objective is that you want to write a program that how to know whether the input number is an even number or an odd number.

So, the first thing here is you need to first understand and think that what can be the logic. So, the logic behind deciding that whether the number is even or odd is that you try to divide the number by 2 and if the remainder comes out to be here 1; then the number is odd and if the number gives the remainder when divided by 2 as 0; that means the number is even. And now, you can recall how you can do it in the R software, you have done the Modulo Division if you try to recall.

So, the modulo division was done by two symbols of percentage sign and this finds the remainder after the division of one number by another. So, now, you want to write this program. So, now, if you try to see; if you use this if else condition, this job become very simple. I will try to write down here if else and then x modulo division equal to 2 by 2

and then if this is exactly equal to 0; that means you are trying to say that you try to divide the number by 2 and if the remainder comes out to be 0, that means the condition is true and under this true condition, you want to write that the number is even number.

And in case if this condition is false; then obviously if the number is not even, it has to be odd, yeah means I am talking only of the this real numbers and integers, right. So, in that case you have to print odd number. So, I try to take here this data vector x equal to 7, 9, 8, 4. Now, you can see here this data vector is operated inside this if else condition and x takes value here 7; it divides by 2 and the remainder comes out to be here 1, which is this equal to 0 false.

So, in case if the condition is false, it will print here odd number; then it will x will take next value 9 and then 9 will be divided by 2, the remainder will come out to be here 1, which is equal to 0, no this is false. So, now, it will print here odd number. Now, the next value here is x equal to 8. So, 8 will be divided by 2, the remainder will come out to be here 0 and 0 equal to 0, yes it is true. So, this will be printed here as a even number.

And finally, x will take the value here 4 and 4 divided by 2; the remainder will come out to be here 0 and 0 is exactly equal to 0 this is true and so the, so it will be printed as the number is even. And you can see here that is what exactly it is happening here; odd number odd number even number even number and because you wanted to print a character, so that is why you have given it inside the double codes, right.

(Refer Slide Time: 27:38)

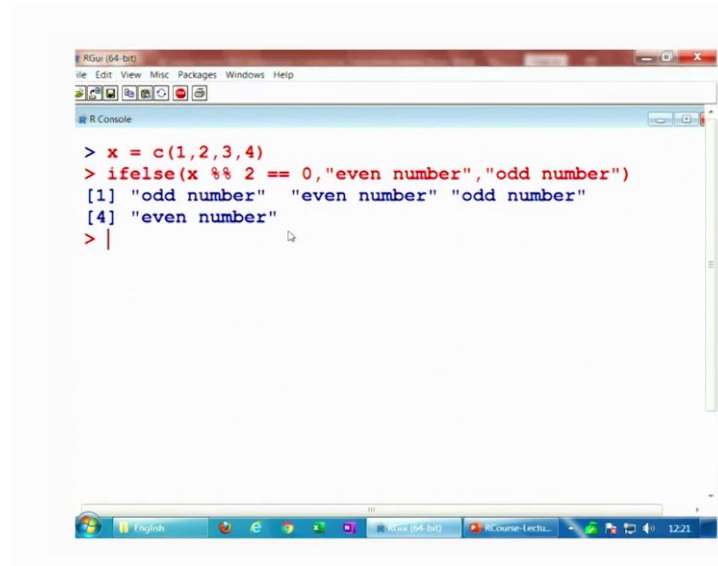
#### 4. Conditional execution: ifelse()

Example 5:

```
> x = c(7,9,8,4)
> ifelse(x %% 2 == 0, "even number", "odd number")
[1] "odd number" "odd number" "even number" "even number"
> |
```

So, that is what is happening here now. And if you try to see here this is the screenshot of the same operation and now I try to show you this thing on the R software also, so that you become more confident, right.

(Refer Slide Time: 27:53)



```
RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
> x = c(1,2,3,4)
> ifelse(x %% 2 == 0, "even number", "odd number")
[1] "odd number" "even number" "odd number"
[4] "even number"
> |
```

So, I tried to copy this condition and I try to take here some number; suppose I try to, I choose here x is equal to say 1, 2, 3 and 4, right. So, you know that 1 and 3 they are odd numbers and 2 and 4 are the even number.

So, if you try to write down here this condition, you can see here this is operated and you get here this outcome. And similarly if you try to take here any other number that is not a very difficult thing for you to identify, right ok. So, now, we come to an end to this lecture and we stop here. So, you can see here that, this was a pretty simple lecture and you have understood the application of two more conditional expectation.

Now, that will be your decision that, in a given problem which control statement you want to use, which conditional execution statement you want to use; whether this is if, if else or something else. And you have to basically think that the use of which of the condition will make the programming easier; because the program should not be unnecessarily very long.

So, these commands have been helped; try to help you in making an efficient program. So, once again I will request you; now this is your turn that you try to think about some



problem and try to first judge that out of these four syntax, which is going to be more useful for you, which of the command is going to help you more.

And then try to write down the program and try to see whether the outcome, which you thought and the outcome which R is giving are they matching; if yes be happy, if not, try to look where is the problem. And one thing I can assure you that, these commands are very helpful when you are trying to do any statistical calculation, mathematical calculations and simulations. So, these are very basic fundamentals, which you need to learn for writing a good program in the future. So, you try to understand it, practice it and I will see you in the next lecture; till then goodbye.