Foundations of R Software

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

Basics of Calculations
Lecture - 22
Loops - for loop

Hello friends, welcome to the course Foundations of R Software and you can recall that in the last couple of lectures we have discussed a couple of topics about the control structures and the control structures are essentially used to control the program or the structure of the program. So, now, continuing on the same lines, in this lecture today we are going to begin with a new topic of control structure that is about loops.

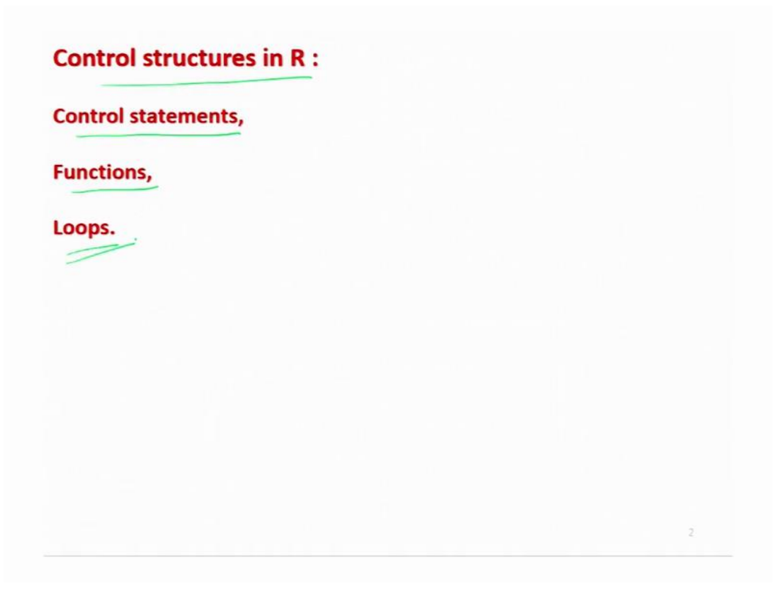
So, the first question comes that, why these loops are needed? Loops are needed whenever you want to repeat the same command more than once. For example, let me try to take a simple example to explain you. Suppose in a class the mark sheet and the grade sheet are to be prepared for all the students and suppose there are 100 students and suppose there are only two classifications that if a student has secured more than 50 percent marks then the student is going to be graded as pass otherwise fail.

So, now that is the rule. So, that command you can write very easily, but now this command has to be repeated over all the student all the 100 students. So, now, first option is that you try to take one student at a time and you try to execute this program and you try to record the outcome and you create the grade sheets.

Second option is that in case if I can have an option that I supply the list of all the students and this program is going to be executed repeatedly for all the students and finally, the grade sheets of all the students will be prepared automatically. So, under these types of situations we try to use the concept of loops.

Now, there are different types of loops. So, in this lecture we are going to talk about the for loop and in the next lecture we are going to talk about the while and repeat loops. So, let us begin our lecture and we try to consider some examples to understand how you can execute the concept of for loop in the R software. So, let us begin our lecture.

(Refer Slide Time: 02:50)



Control structures in R :

Control statements,

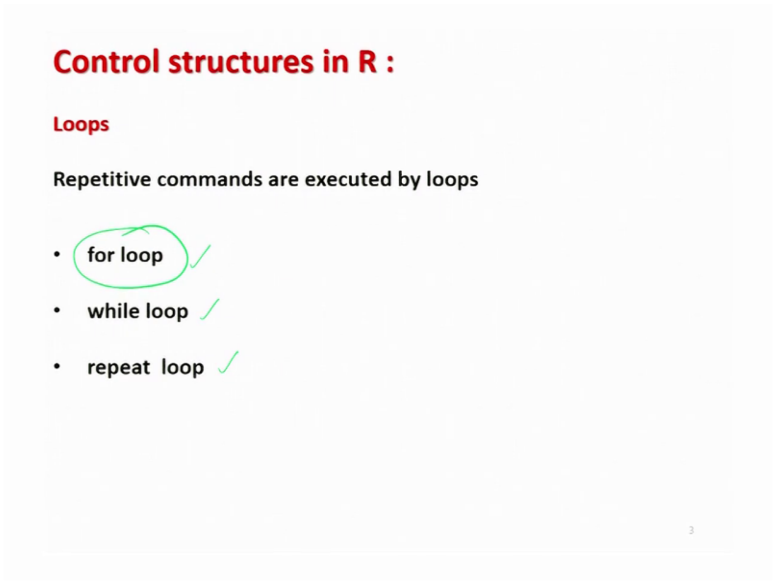
Functions,

Loops.

2

So, you know that we had started a discussion on the control structures in R and we already have done the control statements, conditional executions like if, if else, if else if etc. Then we had considered the some functions and we had considered the function switch and which and now we are coming to the concepts of loops.

(Refer Slide Time: 03:17)



Control structures in R :

Loops

Repetitive commands are executed by loops

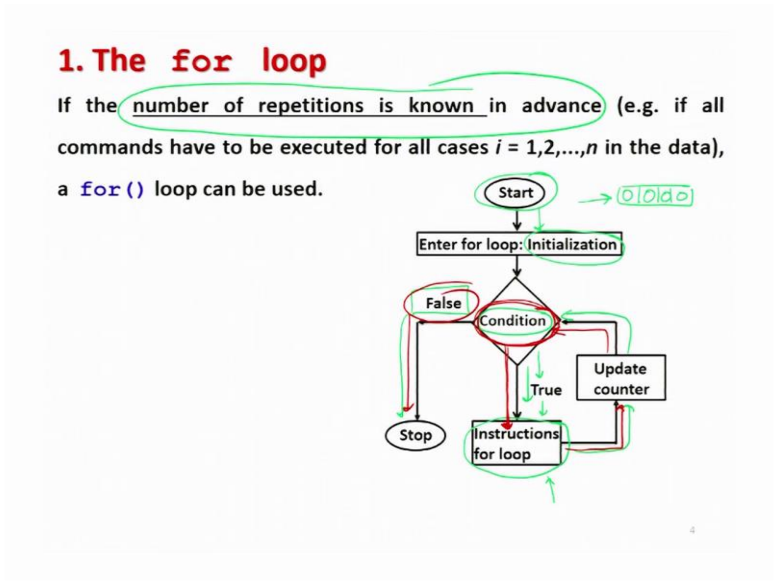
- for loop ✓
- while loop ✓
- repeat loop ✓

3

So, why these loops are needed? These loops are needed to repeat the commands, right that in case if you want to execute a particular command or a set of commands for more than once, then you try to use the concept of loop and there are three popular loops which are popularly called as for loop, while loop, repeat loop.

So, in this lecture we are going to talk about the for loop and all the three loops, they are used in different types of conditions that depending on the need and requirement you can choose whichever loop you want to use. So, I will try to explain you their conditions under which they are going to be used.

(Refer Slide Time: 04:03)



1. The `for` loop

If the number of repetitions is known in advance (e.g. if all commands have to be executed for all cases $i = 1,2,\ldots,n$ in the data), a `for()` loop can be used.

First, let us try to understand about the, for loop. So, the for loop is used when the number of repetitions is known in advance; that means, you know that for how many times you want to repeat the loop. For example, you just consider the example where we know that there are 100 students and we want to prepare their grade sheets. So, whatever is my program, whatever are the commands they have to be repeated 100 times.

So, under such a case, we try to use the concept of, for loop and the flow chart of this for loop is like this, that we start and we enter into the loop. So, whenever you are trying to start the loop you have to first initialize it means, initially something like setting the counter at 0 from or the setting the counter at a point from where you want to measure.

For example, suppose I want to know that I am driving my car and I want to know that how many kilometres I am going to drive today. So, what will I do? At the time of a start, there is a counter and there I will make the reading to be 0 0 0 0 and after I have driven the car for the whole day then I can see that how many kilometers I have driven.

So, this process of making the counter to be at initial value 0 0 0 0 this is called as initialization. So, you know that how the counter is going to work right because in the loops also you have to define a counter which is going to control the number of times the loop is going to be repeated.

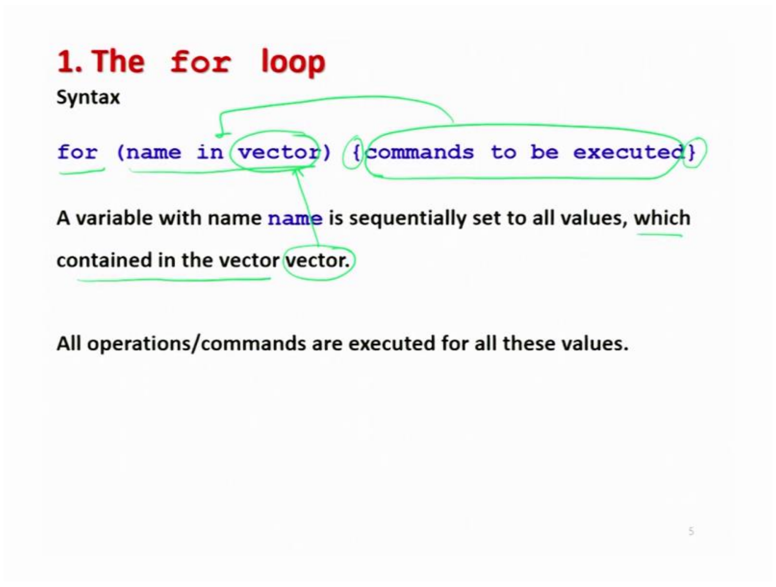
So, now we enter into the loop and yes certainly there will be some conditions which we have to check and based on the outcome of this condition whether the conditions are true or false, we are going to execute the conditions. So, in case if the conditions are true then whatever are the instructions whatever are the commands they are going to be executed and after that and then in order to inform the R software that it has to go further the counter will be updated here.

And this information will be supplied to the condition and once again in case if the condition is going to be true, then once again this process will start and the instructions will be executed and this process will continue. As soon as the condition become false then the program will stop.

So, this is how the for loop works is just like this that the program will start and the program has to be repeated for say 100 students. So, the program will come here and the conditions are going to be checked. In case if the condition is true then whatever is the grade of the student that is going to be reported, then after that the control will come to the next student by updating the counter.

And the next student records will be taken as input and then the condition for the pass of fail is going to be checked and then based on that the instructions are going to be executed and this counter will be updated till we reach to the last student that is a 100th student and as soon as the counter comes to the 101, the condition will become false and then the process will stop. So, this is the way the for loop works.

(Refer Slide Time: 07:51)



1. The for loop

Syntax

```
for (name in vector) {commands to be executed}
```

A variable with name name is sequentially set to all values, which contained in the vector vector.

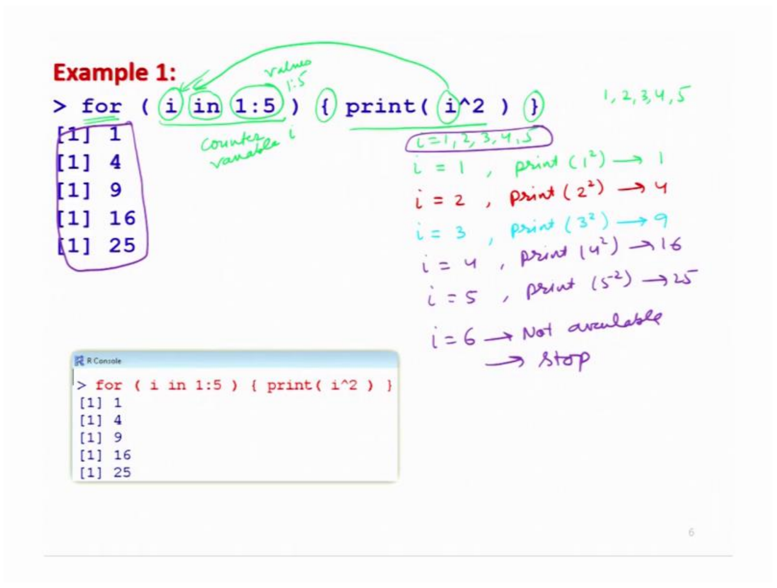
All operations/commands are executed for all these values.

So, now the question is that how are you going to work with the, for loop in the R software? So, the syntax for the for loop is that first you would try to write f o r for and then you have to give here a name inside the parenthesis in the form of a vector and yeah it means you have to be careful that I am writing here, which contained in the vector vector so, this vector is corresponding to this vector, right.

So, we have to give here a counter variable. So, that is going to control the values in this vector and after that whatever commands we want to execute, they are going to be mentioned just after that inside the curly brackets and then the loop will start and these commands are going to be executed for all the values which are given in the counter variable whose information to repeat is stored in the vector.

So, this is how actually for loop works. Do not worry as soon as I take some examples this concept will become very clear.

(Refer Slide Time: 09:03)



Example 1:

```
> for ( i in 1:5 ) { print( i^2 ) }
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
```

```
> for ( i in 1:5 ) { print( i^2 ) }
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
```

So, why not to take an example and try to understand what it does? Suppose I want to print the numbers 1 to 5 1, 2, 3, 4 and 5.

So, in order to do it I try to use the concept of loop and I write down my, for loop as follows for now this is here the counter variable. Counter variable means this is going to control the counter of the number of times the function is the functions are going to be repeated. So, I have to define here a variable this is the counter variable i which is going to inform the for loop that how many times or for what values this for loop has to be repeated right.

So, then I write here after informing the counter variable I try to write down here n and then I try to write down here the values for which I want to repeat the function or the commands. Here because I wanted to repeat this value for the values 1, 2, 3, 4, 5. So, I have written here as a 1 colon 5 otherwise you can give here any value right that can be also in the form of theta vector and then I give the instructions inside the curly bracket that print i square what is this i? This is coming from here right.

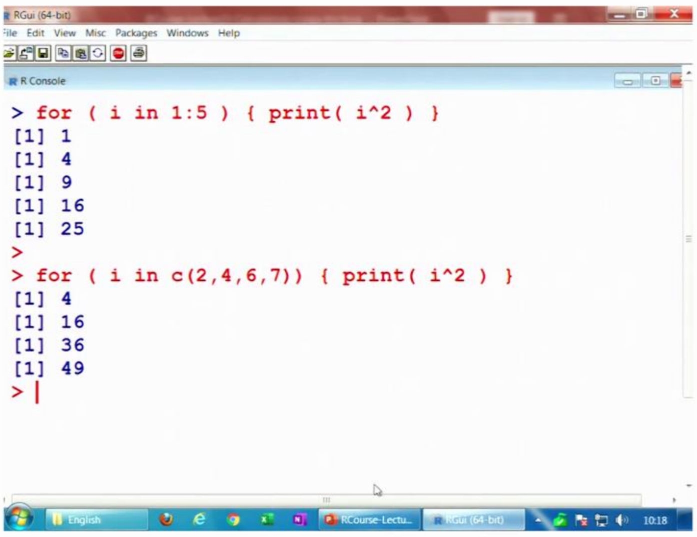
So, in case if you try to see what will happen? As soon as you try to execute it on the R software the control will come to here i and it will try to choose the first value. So, i is going to take here 5 values 1, 2, 3, 4, 5. So, first the value of i equal to 1 is going to be considered and then this value of i will be coming inside the curly brackets and all the instructions to be executed are read.

Then it is here print 1 square. So, this will print here 1. Now after this the second value or the next value in the counter which is i equal to 2 this is going to be considered and it will try to execute the commands inside the curly bracket as print 2 square and this will give you a result 4 and similarly the next value in the i which is here i equal to 3 that is going to be considered and then the control will come inside the curly bracket and whatever are my instructions like as to print 3 square will be executed and we will get here the value 9.

Similarly, it will come to i equal to 4 and it will print here 4 square which is here 16 and then it will come to here i equal to 5 and then it will print here 5 square 25 and now after this there are no more values in this vector in this set of values. So, now I will it will go to i equal to 6, but this is not available.

So, the program will stop here and this is how these things work and you can see here this is here the outcome which you are going to get when you are trying to execute it on the R console, right. So, before going further let me try to show you this command that how it works in the R software inside the R console.

(Refer Slide Time: 12:41)



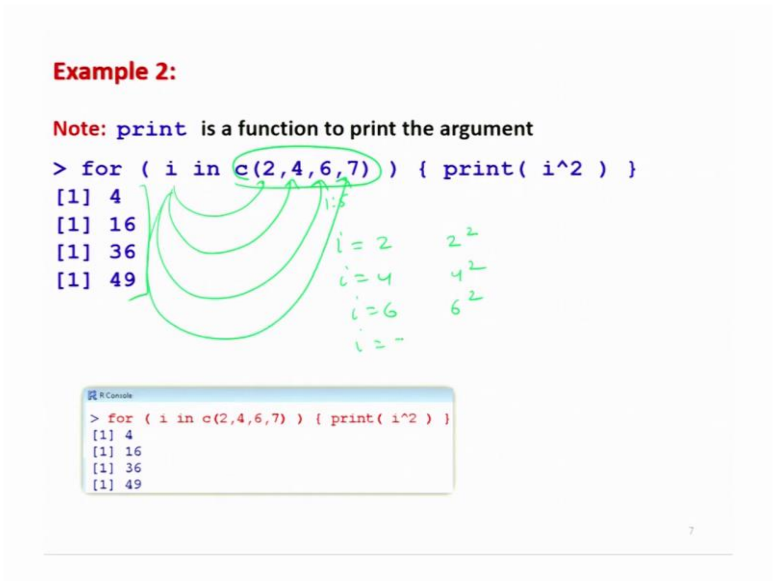
```
> for ( i in 1:5 ) { print( i^2 ) }
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
>
> for ( i in c(2,4,6,7)) { print( i^2 ) }
[1] 4
[1] 16
[1] 36
[1] 49
> 
```

So, now I try to use this command on the R console and you can see here that as soon as you use this command and press enter, it will give you this outcome. Now at the same place I will try to show you that in case if you try to replace this 1 to 5 values by some other values which are not really in the sequence.

So, you can see here what I try to do that instead of 1 colon 5 I try to give here a data vector say 2 comma 4 comma 6 comma 7 right and if you try to see here what will happen what do we expect here? That the control will come to here 2 and it will print here 2 square, then the control will come to 4 it will print here 4 square, the control will come to 6 and then it will print here 6 square and control will come to 7 and it will print 7 square which is 49.

So, let me enter and try to see what happens? So, you can see here these are the 2 square, 4 square, 6 square and 7 square these values are obtained. So, you can give here any value in this data vector and those values are going to be considered in the same order in which they have been mentioned inside the data vector, right.

(Refer Slide Time: 13:53)



Example 2:

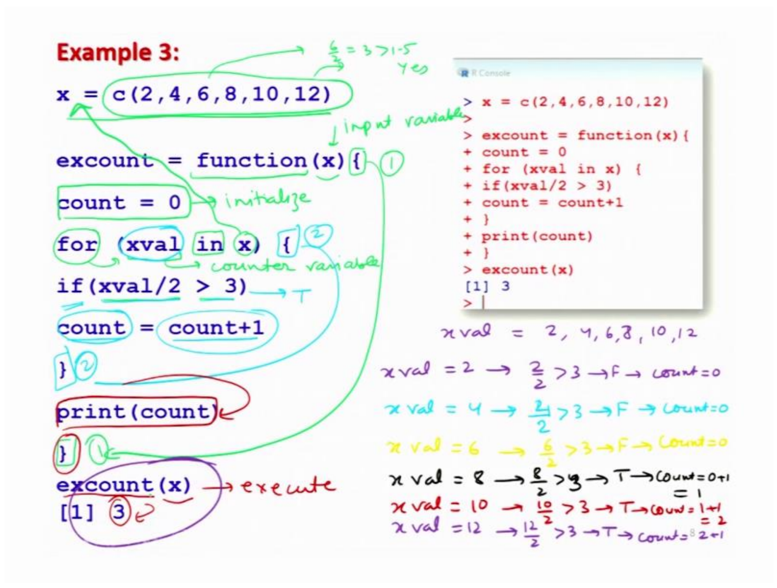
Note: `print` is a function to print the argument

```
> for ( i in c(2,4,6,7) ) { print( i^2 ) }
[1]  4
[1]  16
[1]  36
[1]  49
```

```
> for ( i in c(2,4,6,7) ) { print( i^2 ) }
[1] 4
[1] 16
[1] 36
[1] 49
```

So, yeah this is the same command which I have shown you on the R console. So, I have taken here this data vector here 2, 4, 6, 7 in place of here 1 to 5 and then it is printing here this value, right. So, the same thing is happening that first i is coming to the first value which is i equal to 2 and then it is printing 2 square then after this the next value i equal to 4 is considered and then it is printing here 4 square then the third value is considered i equal to 6 and 6 square is printed and finally, the last value i equal to 7 is considered and 7 square is printed, right, ok.

(Refer Slide Time: 14:41)



Example 3:

```
x = c(2,4,6,8,10,12)

excount = function(x){
count = 0
for (xval in x) {
if(xval/2 > 3)
count = count+1
}
print(count)
}
excount(x)
[1] 3
```

```
> x = c(2,4,6,8,10,12)
>
> excount = function(x){
+ count = 0
+ for (xval in x) {
+ if(xval/2 > 3)
+ count = count+1
+ }
+ print(count)
+ }
> excount(x)
[1] 3
> |
```

6/2 = 3 > 1.5
Yes
input variable
1
initialize
2
counter variable
T
2
1
execute
xval = 2, 4, 6, 8, 10, 12
xval = 2 → 2/2 > 3 → F → count = 0
xval = 4 → 4/2 > 3 → F → count = 0
xval = 6 → 6/2 > 3 → F → count = 0
xval = 8 → 8/2 > 3 → T → count = 0+1 = 1
xval = 10 → 10/2 > 3 → T → count = 1+1 = 2
xval = 12 → 12/2 > 3 → T → count = 2+1

So, now let me try to give you here one more example, yeah. One point where I would request you to be little bit considerate is that here I am going to use here a concept of function, right although I agree that we have not considered this concept up to now we have not understood or I will say that I have not explained you up to now, but that is the topic, which is coming just after the topics of loops.

So, I would request you that you please try to take the this example and after you understand the function you can come back here and then I am sure that you will understand it more easily, but on the other hand this concept is very simple and my objective here is to show you the application of the loop and as I said in the beginning itself that the way we are going to learn this software is that we are trying to take the commands and there is a good possibility that some of the commands, which I am using here may be considered in the forthcoming lectures, but then this is how we have to understand each other and we have to help each other, right.

So, I request you that you please help me and try to understand this example my objective is not at all to explain you the concept of function here, but I will try my best to give you briefly and my main objective is that how this for loop is going to work. So, in this example my objective is that I have got a set of values here, which are contained in the data vector x 2, 4, 6, 8, 10 and 12 and I want to check that in this data vector in case any of the value is and it is divided by 2 that half of the values contained in the data vector is it greater than 1.5.

For example, in case if I say here 6; 6 divided by 2 it is 3 it is greater than 1.5 yes. So, I want to count such values when they are divided by 2 they yield a result or a value which is more than 1.5. So, in order to execute it now you can see here that I have to repeat syntax and commands for the number of times the values are in this data vector and for all the values which are contained inside the data vector.

So, for that I write here function. So, function is only a program means in the common language I can say. So, it is written here as a f u n c t i o n and then inside the parenthesis you try to write down all the input variable or in case if there are more than one input variable that you can also write inside this parenthesis. Then after that whatever commands you want to execute they have to be given inside this curly bracket.

So, let me try to write down this curly bracket number here 1 so, that you can understand it, right. So, whatever you want this has to be written inside these two. So, now I try to give here my commands, which I want to execute. So, what I want here that all the values in the data vector x, they are to be divided by 2 and then they have to be checked whether this is greater than 3 or not right. So, now, for this I use here a loop for loop.

So, for the for loop means, I want to count the values. So, for that I need to define here a counter variable. So, I try to define here a counter variable as a count c o u n t, but you can choose here any name actually whatever you want, right and I initialize it at 0 because the counting will start from 0. In case if you want to start the counting from some other number you can control it here without any problem.

Now, write down my here loop for xval. So, this is my here counter variable. So, this variable is going to control the values inside the x theta vector. So, I try to write down here xval and then in and then my this data vector here x. Now after this I try to enclose all the commands what whatever I want to use inside this curly bracket say like this and here this. So, look let me try to call it here curly bracket number 2 right like this.

So, now my commands are extremely simple I try to use here a conditional execution using if command. So, this if xval divided by 2 is greater than 3, then count will be replaced by count plus 1. So, now, what will happen? This counter variable xval will try to pick up all the values from the data vector x and it will try to use the counter variable count and in case if this condition is true, then it will replace count is equal to count plus 1.

Otherwise it will remain as thus same and after this the program will stop here and then finally, the program will come out of this loop and then it will try to print whatever are whatever is the value of the variable count and then after this function will stop and after that in case if you want to execute this program, then you have to write down the name of the function e x c o u n t and within the parenthesis the input variable x, right and it will give you here the value 3.

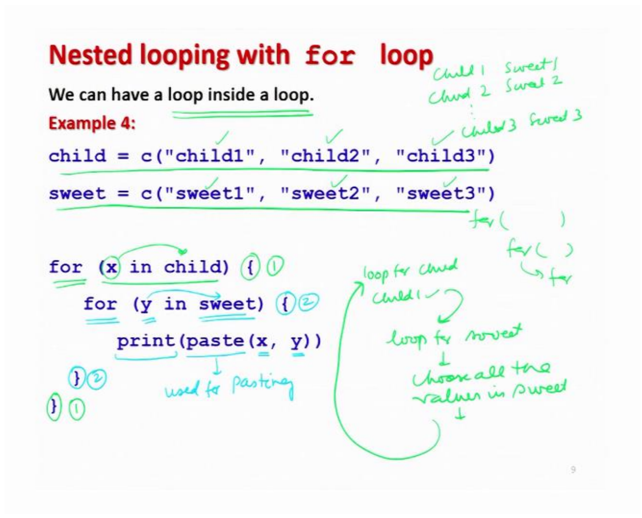
How? Let us try to understand how this is working right. So, in case if you try to see here you are trying to say xval takes the value 2, 4, 6, 8, 10 and 12. So, now, the first value will be chosen xval equal to 2, now 2 divided by 2 is greater than 3, the condition is false. So, this count will be taken as 0 because this is the initial value.

Now, after this the next value in the xval which is equal to here 4 that will be taken and 4 divided by 2 will be considered the answer is 2 is it greater than 3 answer is false and once again the count will remain here as a 0. Now after this the next value in the xval, which is here 6 xval is equal to 6 will be taken in 6 divided by 2 is 3, 3 is greater than 3 this is false and once again the count will assume the value 0.

Now, after this the next value xval equal to 8 will be taken a divided by 2 is greater than 3 answer is true and then the count will be replaced by count plus 1. So, this count will become here 0 plus 1 that is 1. Now after this the next value xval which equal to here 10 that will be taken then the same operation will be done 10 divided by 2 is it greater than 3 yes 5 is greater than 3. So, the condition is true.

So, now after this the count will become here the count plus 1 1 plus 1 is equal to 2 and now finally, what will happen? The last value in the xval, which is equal to here 12 that will be taken 12 divided by 2 is greater than 3 this is true and then the count will become count plus 1 which is here 2 plus 1 equal to here 3 what is given here. So, this is how this loop is going to work in this case, right.

(Refer Slide Time: 23:01)



Nested looping with `for` loop

We can have a loop inside a loop.

Example 4:

```
child = c("child1", "child2", "child3")
sweet = c("sweet1", "sweet2", "sweet3")

for (x in child) {
   for (y in sweet) {
      print(paste(x, y))
   }
}
```

child1 sweet1
child 2 sweet 2
child3 sweet 3
for( )
for( )
for
loop for child
child1
loop for sweet
choose all the values in sweet
used for pasting

So, now similarly I try to take here one more example and after that I will try to show you both these examples on the R console. Now in this case what I want to show you is that we can have a loop inside a loop I mean there can be more than one loop in a function in a program and those loops can be inside the loops, right. So, that is just like you try to write down here a for command and then inside this for command you give here one more for command and it is possible that within this for loop you try to give here one more loop and so on.

So, let me try to explain you with this example right. So, suppose I try to take here 2 data vectors one is here child and another here is sweet. So, in the data vector child there are 3 children child1, child2 and child3 and suppose for the sweet there are 3 sweets sweet1, sweet2 and sweet3 and I simply want to print the combination of the values in the two data vectors child and sweet, right.

So, what I try to do here is like this suppose I try want to print child1 and sweet1, child2 with sweet1 2 and up to here see here child3 and sweet3 like this. So, now, I can use here the concept of for loop and I can define here 2 for loops one for the data vector child and another for say sweet. So, I try to do it here like this.

Please try to understand and then I will try to explain you how it works. I try to take care of for loop and then inside the parenthesis I try to write down here a counter variable, this counter variable is going to work for the data vector child, right. So, that is going to control the values in the vector which are child1, child2 and child3 and I try to write down here this curly bracket. So, let me call it this curly bracket as say here 1.

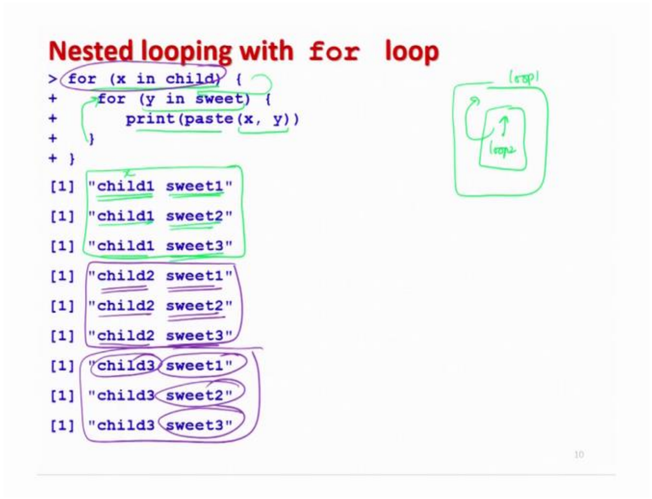
So, whatever I want to write inside this curly bracket that will be executed by this for loop which is related to child. Now within this loop I try to write here one more loop for y in sweet. So, I try to give here one more counter variable y which is trying to control the values in the data vector sweet, sweet1, sweet2 and sweet3 and whatever I want to do this is contained in this curly bracket which is here as number 2.

I simply want to here say paste the values of child and sweet and I want to print them. So, for paste there is a command here paste. So, this is used for guessing the values, ok we have not done this command yet, but we are going to do it in the forthcoming lecture, but as the name suggest paste. So, it is not very difficult to understand.

So, now if you try to see what will happen here? Let me try to try to explain you. So, now, the loop will start here and then it will first try to execute the loop for child. So, there are three values here child1. So, the loop will start it will try to pick up the value of child1 and then it will try to come inside the loop for sweet and it will try to choose all the values in sweet and then it will try to execute the command for printing.

Now, after this when all the values in the sweet are consumed then it will come back once again to the loop or child and it will try to pick up the second value child2 and it will once again it will try to pick up all the values in the data vector sweet and it will try to combine them together.

(Refer Slide Time: 27:21)



Nested looping with `for` loop

```
> for (x in child) {
+     for (y in sweet) {
+        print(paste(x, y))
+     }
+ }
[1] "child1 sweet1"
[1] "child1 sweet2"
[1] "child1 sweet3"
[1] "child2 sweet1"
[1] "child2 sweet2"
[1] "child2 sweet3"
[1] "child3 sweet1"
[1] "child3 sweet2"
[1] "child3 sweet3"
```

10

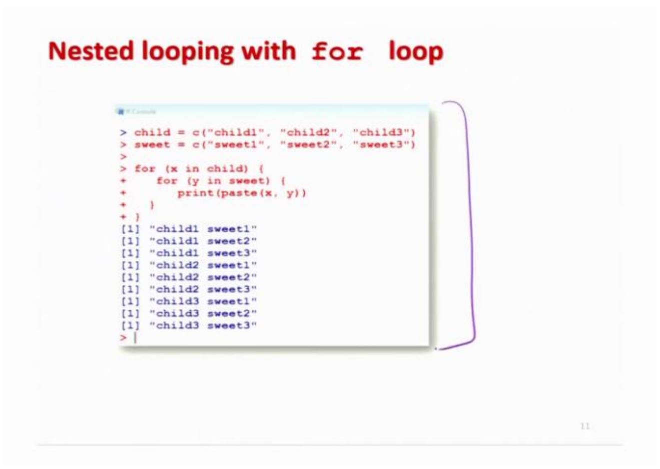
So, let us try to first see what really happens and then I will try to show you what is really happening. So, first of all the control comes here in the x in child and it tries to pick up the first value child1. Then after this the control comes to the second data vector y in sweet and it tries to pick up the first value in the sweet which is sweet1 and then it takes here x and y here as say x as child1 and y here as a sweet1 and it tries to paste and then here print.

Now, after this the control will come here to here y means, you have to understand that the if I have here 1 loop like this and there is another loop here like this inside this loop say loop 1 and then this is here loop 2. So, first it works from inside that whatever are the values in the loop 2 first they are executed and then after execution it will come out of the loop and it will try to choose the values in the loop 1.

So, now that is what is happening. So, it will try to now select all the values in the data vector sweet which are sweet1, sweet2 and sweet3 and it will keep the child1 as fixed, right. Once this is done now the control will come to the for loop which is for the child and it will try to pick up the second value child2 and then it will come to the loop for the y and it will try to pick up the first value in the sweet vector sweet1.

And then it will try to take the second value sweet2 sweet3 and this child2 is going to be remain the same and it will try to paste and then print and the same process will come that finally, it will come to that data vector in x and it will try to choose here the value child3 and then it will come to the data vector in sweet which is here sweet1, then sweet2, then sweet3 and then it will complete the loop like this, right.

(Refer Slide Time: 29:33)

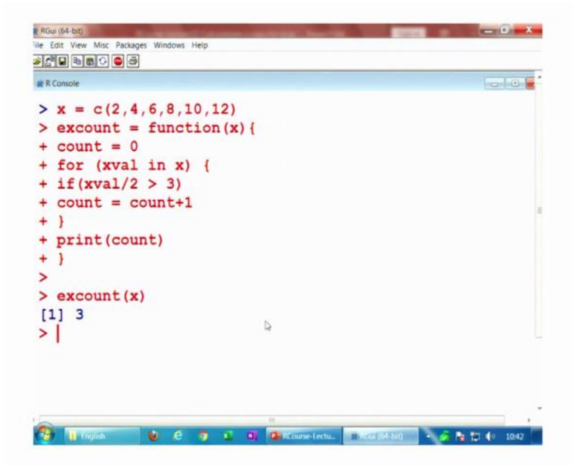


Nested looping with `for` loop

```
> child = c("child1", "child2", "child3")
> sweet = c("sweet1", "sweet2", "sweet3")
>
> for (x in child) {
+     for (y in sweet) {
+         print(paste(x, y))
+     }
+ }
[1] "child1 sweet1"
[1] "child1 sweet2"
[1] "child1 sweet3"
[1] "child2 sweet1"
[1] "child2 sweet2"
[1] "child2 sweet3"
[1] "child3 sweet1"
[1] "child3 sweet2"
[1] "child3 sweet3"
> |
```

So, this is how the loop is going to work and you can see here this is here with the screenshot of the same operation, right. So, let me try to show you first these examples on the R console so, that you get here more confidence and then I will try to do some other execution. So, if you try to see here I try to choose here this vector here x which is here like this and then I try to just copy and paste the command to save my time.

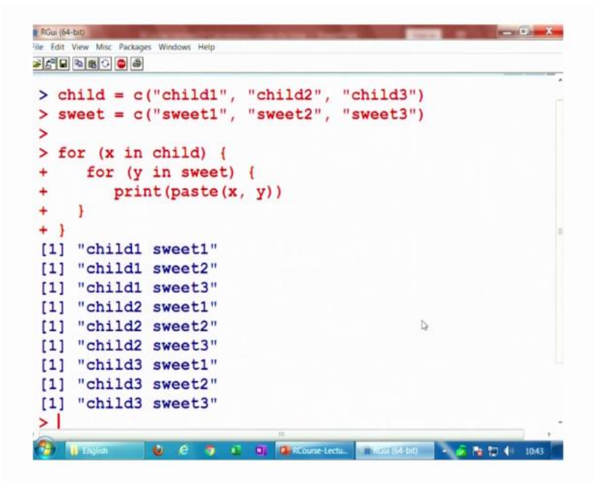
(Refer Slide Time: 29:58)



```
> x = c(2,4,6,8,10,12)
> excount = function(x){
+ count = 0
+ for (xval in x) {
+ if(xval/2 > 3)
+ count = count+1
+ }
+ print(count)
+ }
>
> excount(x)
[1] 3
> |
```

So, if you try to see here it is like this and if you want to execute it I try to write down here x count and then here if I try to write down here x, you can see here this is giving me the value 3, right. So, this is how the program will work without any problem and similarly I try to choose here these two data vector and I also try to choose this command.

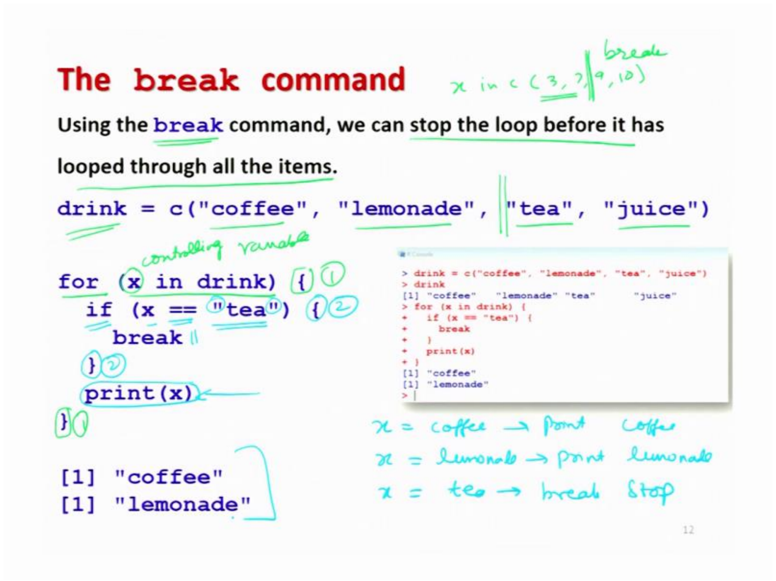
(Refer Slide Time: 30:36)



```
> child = c("child1", "child2", "child3")
> sweet = c("sweet1", "sweet2", "sweet3")
>
> for (x in child) {
+    for (y in sweet) {
+       print(paste(x, y))
+    }
+ }
[1] "child1 sweet1"
[1] "child1 sweet2"
[1] "child1 sweet3"
[1] "child2 sweet1"
[1] "child2 sweet2"
[1] "child2 sweet3"
[1] "child3 sweet1"
[1] "child3 sweet2"
[1] "child3 sweet3"
> |
```

So, I simply try to paste it here you can see here that the outcome is coming out to be here like this, right. So, you can see here means executing these commands is not a difficult at all, right. So, now let me try to give you some more options, which are useful in using the loop.

(Refer Slide Time: 30:56)



The break command

x in c(3, 7, 9, 10) break

Using the break command, we can stop the loop before it has looped through all the items.

```
drink = c("coffee", "lemonade", "tea", "juice")

for (x in drink) {
  if (x == "tea") {
    break
  }
  print(x)
}

[1] "coffee"
[1] "lemonade"
```

controlling variable

```
> drink = c("coffee", "lemonade", "tea", "juice")
> drink
[1] "coffee"   "lemonade" "tea"      "juice"
> for (x in drink) {
+   if (x == "tea") {
+     break
+   }
+   print(x)
+ }
[1] "coffee"
[1] "lemonade"
> |
```

x = coffee $\rightarrow$ print coffee

x = lemonade $\rightarrow$ print lemonade

x = tea $\rightarrow$ break stop

12

So, these are some additional commands which are required. So, for example, let me try to first consider the command here break b r e a k this command is used to stop the loop before it has loop through all the items.

Suppose, you want to stop it somewhere in between of the data vector then you have to use the command loop, right. So, for example, if you are trying to say here x in c say 3, 7, 9 and 10 and you want to in break it here that the program should work only for 3 and 7 and not for 9 and 10. So, here you have to give the command here break.

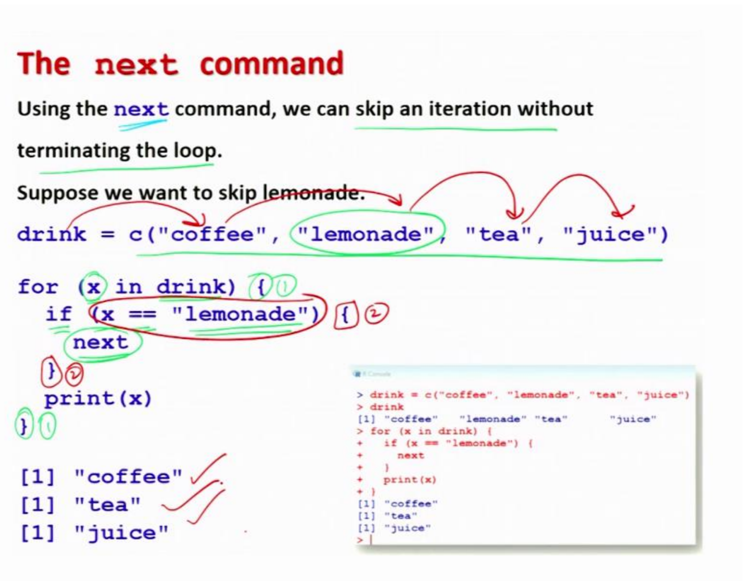
So, how to get it done in the, for loop? So, look let me try to explain you with this a small example suppose I try to take here the vector get vector drink and it has four values coffee, lemonade, tea and juice, right and I want to execute this program in such a way that I want to print this value print coffee, print lemonade, print tea and print juice, but I want to stop it at tea that as soon as the control comes to here tea the function or the program could stop.

So, what I try to do here that I try to write down here the, for loop as for x in drink. So, x is going to be my controlling variable controlling this variable means it is going to control the values in the drink vector. So, it will try to this and then I try to give my commands inside this a curly bracket let me try to give it here a number 1 and then I try to write my condition if x is logical equal to tea, right.

So, you have to give it inside the double quotes, then you try to execute these commands which are inside this curly bracket number 2 and what is that command? Break that means, break the program, and then whatsoever values you have collected before that try to paint them right ok.

So, now if you try to see what will happen? The control will start for x it will take the first value here coffee and it will print here coffee. Now the value will come to the next value in the data vector drink it will try to take here lemonade and then it will try to print lemonade and then it will try to come to the third value which is here tea, but the command here is that if you if the value in the x is exactly equal to or logically equal to tea then break. So, it will stop here. So, you can see here that this will be the outcome coffee and lemonade, right.

(Refer Slide Time: 34:08)



The next command

Using the next command, we can skip an iteration without terminating the loop.

Suppose we want to skip lemonade.

```
drink = c("coffee", "lemonade", "tea", "juice")

for (x in drink) { 1
  if (x == "lemonade") { 2
    next
  } 2
  print(x)
} 1

[1] "coffee"
[1] "tea"
[1] "juice"
```

```
> drink = c("coffee", "lemonade", "tea", "juice")
> drink
[1] "coffee"   "lemonade" "tea"      "juice"
> for (x in drink) {
+   if (x == "lemonade") {
+     next
+   }
+   print(x)
+ }
[1] "coffee"
[1] "tea"
[1] "juice"
> 
```

So, after this break command there is another command here next n e x t. So, this command is going to use when we want to skip an iteration without committing the loop right. For example, in the same example if I consider suppose I have these four values coffee lemonade tea and juice and suppose I want to execute the same program for all the values except for lemonade, right.

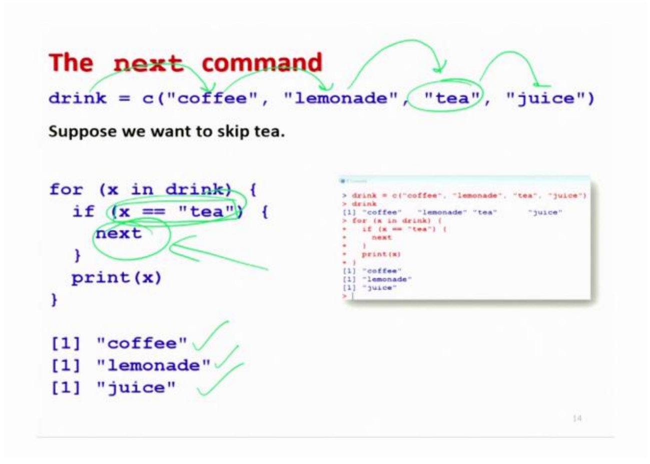
So, what I will do? That I will try to write down the same program, but as we had used here this here break I will try to use here next. So, what will happen here? The loop will start and it will try to choose all the values in the drink using the variable x and now you are trying to say that whatever is written inside this curly bracket number 1 that is going to be executed. So, now, the and the commands are if x is exactly or logically equal to lemonade then say next right. So, that is going to happen here.

So, now in case if you try to see here what will happen here? Yeah, first you have to just see that here also is here this curly bracket also. So, yeah do not forget about this curly bracket this is here the curly bracket number 2. So, whatever you are writing within this curly bracket that is going to be there.

So, now in case if you try to see here, what will happen? The program will start it will take the first value in the vector drink coffee it says no issue and it will print here coffee. Now after this it will try to jump to the next value lemonade, but it finds here that the program is saying that when x is equal to lemonade then next do not execute it.

So, now it will escape and then it will come to tea it will print here tea and then after that it will choose next value juice and it will try to print here juice. So, that is how actually this works right and simply we have here simply I try to suppose repeat this example for a different value.

(Refer Slide Time: 36:36)



The next command

```
drink = c("coffee", "lemonade", "tea", "juice")
```

Suppose we want to skip tea.

```
for (x in drink) {
  if (x == "tea") {
   next
  }
  print(x)
}

[1] "coffee"
[1] "lemonade"
[1] "juice"
```

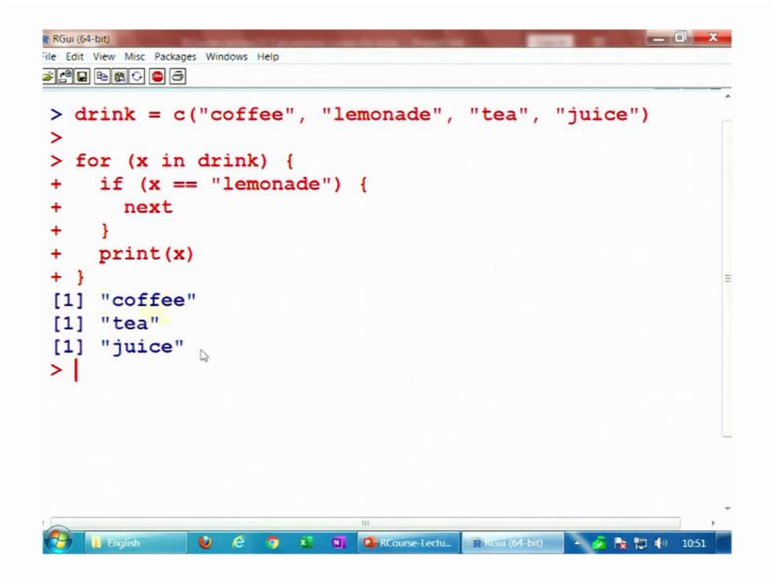
```
> drink = c("coffee", "lemonade", "tea", "juice")
> drink
[1] "coffee"   "lemonade" "tea"      "juice"
> for (x in drink) {
+   if (x == "tea") {
+     next
+   }
+   print(x)
+ }
[1] "coffee"
[1] "lemonade"
[1] "juice"
> 
```

14

Suppose, I want to skip tea so, in the same program I try to write down here if x is exactly equal to tea then please skip and use the command next for doing it. So, the program will start from a drink it will take the first value coffee there is no issue in coffee. So, it will try to print here coffee, then it will come to here lemonade, then it will print here lemonade there is no issue after that it will come to here tea now it is saying that ok if x is equal to tea then next. So, it will skip it and then it will come to juice and it will try to print here juice.

So, you can see here this is how this program is going to work, right. So, let me try to show you these things on the R console and so that you get here more confident. So, I try to copy these values and I try to execute it here you can see here.

(Refer Slide Time: 37:43)

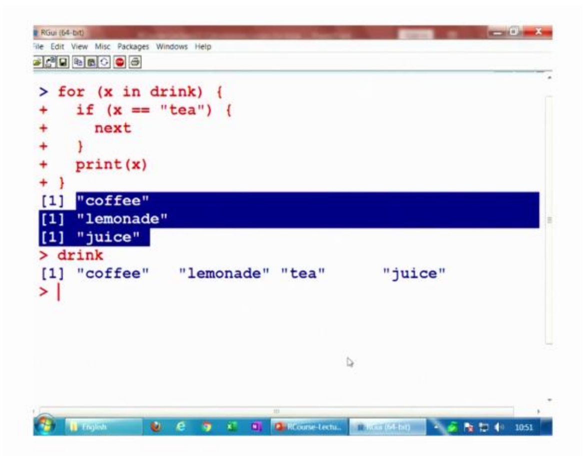


```
> drink = c("coffee", "lemonade", "tea", "juice")
>
> for (x in drink) {
+   if (x == "lemonade") {
+     next
+   }
+   print(x)
+ }
[1] "coffee"
[1] "tea"
[1] "juice"
> 
```

So, drink here is like this and I try to use the, for loop here where I am trying to give a break when x is equal to tea. So, the first two values coffee and lemonade which are before tea they are printed here as a coffee and lemonade. And similarly if you try to see here I try to take here another command and I want to escape here the lemonade.

So, what will happen? The same data vector, but I am trying to use here the command next and when x is lemonade. So, except lemonade all the things will be used here and similarly in case if you want to escape the tea, then you have to just replace x equal to tea and then you can see here this it would be like here and this is your here the data vector drink.

(Refer Slide Time: 38:33)



```
> for (x in drink) {
+   if (x == "tea") {
+     next
+   }
+   print(x)
+ }
[1] "coffee"
[1] "lemonade"
[1] "juice"
> drink
[1] "coffee"    "lemonade" "tea"       "juice"
> 
```

So, you can see here the tea is skipped and all coffee lemonade and juice they are printed here. So, now after this we come to an end to this lecture. Well, you can see that in this lecture we have covered only one simple command, but it was long why? Because this is the first time you are going to understand the concept of the loop in the R software.

So, that is why I had to explain you all the basic functioning's because unless and until you understand how R is going to work for the loops you cannot use it for your own work. So, that is why I had tried my best to explain you in the quite a lucid way in detail. The functioning is very simple, the command is very simple the only thing is this depending on the need and requirement you have to use for loop with different types of option right break or next or a command, which is a combination of them and so on.

So, I would request you why do not you take some example and try to practice it try to see what you really you want to do. Suppose for example, you try to take more than two for loops and try to give some values two or three values and see that whether the outcome is coming is it matching with what you wanted means, I have taken in two loops all the three values you can take two values in the loop number 1 and say three values in the loop number 2 and you can means a local number of values in the two loops and try to see how do you get this outcome?

You can use here different types of command like some product etc., which you have done well I have used here the command here paste and print, but anyway you understand them. So, you try to practice it and I will see you in the next lecture, till then goodbye.