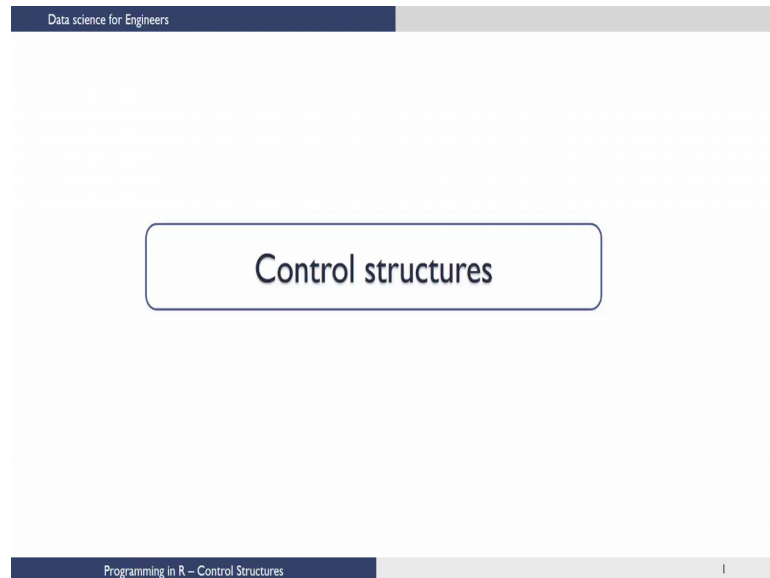


**Data Science for Engineers**  
**Prof. Raghunathan Rengaswamy**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Madras**

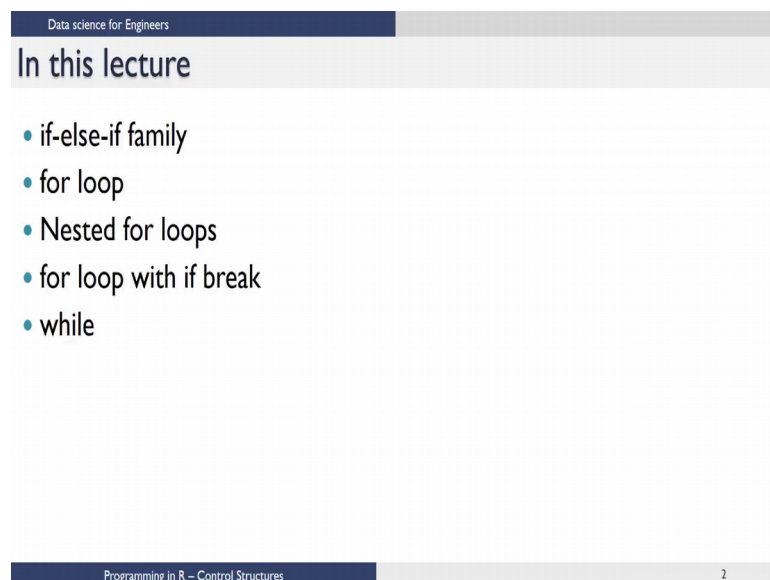
**Lecture - 10**  
**Control structures**

(Refer Slide Time: 00:12)



Welcome to lecture 9 in the R module of the course, data science for engineers. Here we will look at the control structures in R.

(Refer Slide Time: 00:25)



In this lecture we are going to talk about if else if family constructs, for loop nested for loops, for loop with if break and while loop. Control structures can be divided into 2 categories.

(Refer Slide Time: 00:38)

Data science for Engineers

## Control structures

- Execute certain commands only when certain condition(s) is satisfied (if-then-else)
- Execute certain commands repeatedly and use a certain logic to stop the iteration (for, while loops)

Programming in R – Control Structures

The first one is where you need to execute certain commands only when certain conditions are satisfied and example of this control structure is if then else type of constructs. The second one is execute certain commands repeatedly, and use certain logic to stop the iteration examples for this kind of constructs are for and while loops.

(Refer Slide Time: 01:04)

Data science for Engineers

## If else family of constructs

If, If else and If-elseif - else are a family of constructs where:

- A condition is first checked, if it is satisfied then operations are performed
- If condition is not satisfied, code exits construct or moves on to other options

| If construct                                                                                                        | If-else if-else construct                                                                                                                         |
|---------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>if (condition) {<br/>  statements<br/>}</pre>                                                                  | <pre>if (condition) {<br/>  statements<br/>} else if (condition) {<br/>  alternate statements<br/>} else {<br/>  alternate statements<br/>}</pre> |
| <p>If-else construct</p> <pre>if (condition) {<br/>  statements<br/>} else {<br/>  alternate statements<br/>}</pre> |                                                                                                                                                   |

Programming in R – Control Structures

First look at if construct, what if does is if checks for a condition if the condition is satisfied it will execute the statements that are in the if loop. The next level is if else construct, where we want to do certain operations if the condition is satisfied and if not, we want to do some alternate operations.

So, the if else construct syntax looks like this, if the condition is satisfied perform this statements else perform this alternate statements. The next level is if else if else construct. So, here you will have 2 things if a condition is satisfied execute this statement; else if you check for another condition if that condition is satisfied execute this alternate statements, if both of them are not satisfied then do something else so the syntax is as follows; to illustrate this if let us consider this example here.

(Refer Slide Time: 02:09)

Data science for Engineers

### If else family of constructs: Example

IN THE EXAMPLE BELOW:

- IF x is greater than 7 operations inside the curved braces would occur
- Else the next condition i.e.  $x > 8$  would be checked, if this too is not true
- Final else condition is checked and if that too is false, there is no change in 'x'

| Code                                                                                                      | Console Output                                                                                                                                    |
|-----------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre># If-elseif - else example x=6 if(x&gt;7){   x=x+1 }else if(x&gt;8){   x=x+2 }else {   x=x+3 }</pre> | <pre>&gt; # If-elseif - else example &gt; x=6 &gt; if(x&gt;7){ +   x=x+1 + }else if(x&gt;8){ +   x=x+2 + }else { +   x=x+3 + } &gt; x [1] 9</pre> |

Programming in R – Control Structures

So, we have assigned a value of 6 to x, we are checking if x is greater than 7 because your value is 6 and we are checking a condition 6 is greater than 7 which is false this statement will not be executed. Now it will check whether the next condition, x is greater than 8; again this condition also fails because 6 is not greater than 8 and this part is not executed, since this 2 parts are not executed it will move to the else and then it will increment the value of x by 3; that means, you have now x as 6, 6 plus 3 is 9 and the value of 9 is assigned to x and this piece of code is executed in R you can see that the output is 9.

(Refer Slide Time: 03:00)

Data science for Engineers

## Sequence function

- A sequence is one of the components of a 'for loop'
- Sequence function syntax : `seq(from, to, by, length)`
- Creates equi-spaced points between 'from' and 'to'

| Parameter | Description                    |
|-----------|--------------------------------|
| from      | starting number                |
| to        | ending number                  |
| by        | increment or decrement (width) |
| length    | Number of elements required    |

Console Output

```
Console ~ ↻  
> seq(from=1, to=10, by=3)  
[1] 1 4 7 10  
> seq(from=1, to=10, length=4)  
[1] 1 4 7 10  
> seq(from=1, to=10, by=4)  
[1] 1 5 9  
> |
```

Programming in R – Control Structures



Next, we move on to the for construct to understand the for function we need to understand what is a sequence function first. So, let us see what is a sequence function, sequence is one of the components of the for loop that is the reason why we are looking at the sequence function now, sequence function syntax is as follows.

Sequence function contains from the starting number from which the sequence has to begin to the ending number with which the sequence has to begin, you can define the sequence by either providing the by or length, when you provide this argument by it will specify by what increment or decrement the sequence has to be generated, when you provide this argument length. So, what it does is it will create number of elements that are required from the starting number to the ending number you can see the examples here, let us now assume that I want to create a sequence from 1 to 10 and then I want the width of 3.

So, the argument which I want to pass is by equal to 3, this will create one separated by 3 and then 4 4 and it leaves again 3 values and then 7 and then it leaves another 3 values and then a 10 instead I can do the same by specifying the length. So, the way you can do that is as follows. I want to generate a sequence from 1 to 10 which contains the 4 elements. So, it will generate the same thing so it will take from one and start from 1 and and go up to 10 which contains 4 elements. Now if I want to say I want to create a sequence from 1 to 10 which is having a width of 4 this is how the output looks.

(Refer Slide Time: 04:50)

## for loop, Nested for Loops

The structure of a for loop construct comprises:

- A 'sequence' which could be a vector or a list
- 'iter' is an element of the sequence
- Statements

Nested for-loop : one or more for loop constructs are located within another.

For loop construct

```
for(iter in sequence) {
  statements
}
```

Nested For loop construct

```
for(iter1 in sequence1) {
  for(iter2 in sequence2) {
    statements
  }
}
```



Now let us move on to the for loop the structure of for loop construct comprises of a sequence which could be a vector or a list an iter which is an element of the sequence and the statements that are needed to be executed.

So, if you see the structure of this for loop, iter in a sequence as we seen iter is an element in the sequence the sequence is a list R vector; for every element in this sequence execute this statements is what this for loop construct is saying. So, next level of the for loops is a nested for loop. When you say nested for loop it means we have one or more for loop constructs that are located within another for loop.

The structure of the nested for loop is as follows, the for loop here is an inner for loop and the for loop, outside is called outer for loop for every iter 1 in the sequence 1 this for loop will get executed , it will go to the for loop 2 where it will perform this operations on sequence 2 for every iter 2 and return the output to illustrate this for loop.

(Refer Slide Time: 05:58)

Data science for Engineers

## for loop: Example

Open a script file and type the following statements. Save the file as "forloop" and execute the script file

The value of 'sum' keeps changing inside the loop

```
1 n=5
2 sum=0
3 for(i in seq(1,n,1)){
4   sum = sum + i
5   print(c(i,sum))
6 }
```

Initializing sum=0  
loop variable

| Loop variable (i) | sum |
|-------------------|-----|
| 1                 | 1   |
| 2                 | 3   |
| 3                 | 6   |
| 4                 | 10  |
| 5                 | 15  |

Programming in R – Control Structures



We will have an example here, I am initialising number of elements to be 5 and some to be 0. I am having a sequence which is starting from 1 and then ending at 5 with a width of 1, what I am doing inside the for loop is I am assigning sum plus i value to the variable sum inside this for loop and I am printing the i which is the iter value in the loop and the sum when you execute this function this is how it behaves.

So, you have n is equal to 5 it will keep it in memory. So, when we execute and sum is equal to 0 it will initialize the sum to 0, for the first time it enters into the loop it will take value of 1 from the sequence. And then you have already sum as 0, 0 plus 1 is 1 and it will assign value 1 to the sum. You can see that in the first iter or a first iteration the value of sum is 1. In the second iteration you have the value sum as one it will go to the next iteration; that is now the i value is 2. So, you have already sum value as one and i is 2, 1 plus 2 is 3 and the value of 3 is assigned to the variable sum in the second iteration, you can see that value of the sum is 3 here and so on.

Since the sequence runs up to 5 the sum will be 15 at the end of 5 iterations, sometimes it is necessary to stop the function when you feel that the required condition is satisfied.

(Refer Slide Time: 07:41)

Data science for Engineers

## For- loop with if-break

A "break" statement once executed, program exits the loop even before iterations are complete  
"break" command comes out of the innermost loop for nested loops

```
1 n=100
2 sum=0
3 for(i in seq(1,n,1)){
4   sum = sum + i
5   print(c(i,sum))
6   if(sum>15){
7     break
8   }
```

| Loop variable (i) | sum | Condition (sum>SUM) |
|-------------------|-----|---------------------|
| 1                 | 1   | False               |
| 2                 | 3   | False               |
| 3                 | 6   | False               |
| 4                 | 10  | False               |
| 5                 | 15  | False               |
| 6                 | 21  | True                |

break" command, the loop is terminated after the 6<sup>th</sup> iteration

Programming in R – Control Structures



This can be achieved using a break statement in the for loop. So, let us see how to do this I am assigning a variable value of 100 to the variable n and I am initializing the sum as 0. Now I want to stop the loop when the sum exceeds 15, how do I do that? So, in the for loop what you have to have is if break construct. So, in the for loop these are the statements that are available for every iteration I am adding this sum and then iter value and assigning it to the sum, and I am printing the vector which is containing the loop variable and sum.

And I have to check a condition if the sum is greater than 15 I will say break because this is the condition which I want to break the loop. So, this break statement once executed the program exit the loop even before the iterations are complete. So now, let us see how this things work so in the first iteration the loop variable has a value 1 because it is a sequence starting from 1 and the last value is 100 here. So, we have seen in the previous example also for the first time the value of the sum becomes 1.

And it checks the condition if sum is greater than 15 because sum value is 1 it is not greater than 15 the break statement will not be executed. And the break statement will not be executed until 5 iterations when the iteration number 6 comes sum value is already 15, and the iteration value now is 6, 15 plus 6 will become 21 and that 21 will get assigned to the variable sum.

Now if this condition is checked if sum is greater than 15; this condition is satisfied and the break statement get executed; once this break statement get executed the program quit from the for loop next we move on to another construct which is while loop.

(Refer Slide Time: 09:43)

Data science for Engineers

While loop

A while loop is used whenever you want to execute statements until a specific condition is violated

Consider the sequence of natural numbers.

What is the value of the natural number up to which the calculated sum is less than specified "Fin\_sum"?

$$1+2+3+...+n = \text{Fin\_sum} (15)$$

```
1 sum=0  
2 i=0  
3 Fin_sum=15  
4 while(sum<Fin_sum){  
5   i=i+1  
6   sum=sum+i  
7   print(c(i,sum))  
8 }
```

Initialize the variables

| Loop variable (i) | sum | Condition (sum<Fin_sum) |
|-------------------|-----|-------------------------|
| 1                 | 1   | True                    |
| 2                 | 3   | True                    |
| 3                 | 6   | True                    |
| 4                 | 10  | True                    |
| 5                 | 15  | False                   |

Programming in R – Control Structures



A while loop is used whenever you want to execute statements until a specific condition is violated, we can see it as an akin to for loop with if break construct.

Let us consider a sequence of natural numbers; you want to find a natural number n after which the sum of the natural numbers n is greater than certain final sum you wanted to be. You will consider the same example as we have seen I am initializing the sum as 0 and the initial variable i is equal to 0. And I want the final sum to be 15. This is how I write a while function while sum is less than final sum. I want to increment the i value by 1 and then reassign it to the value i and I also want to increase the sum by the iter value and then reassign it to sum and then finally, print the iteration value and the sum. These commands get executed until the loop variable has a value 4.

Let us understand how does it works; for the first time i is 0 it will check the condition sum is less than final sum, the condition is true what it does is it will increment the i by 1. So, i plus 1 which is 0 plus 1 you will get the loop variable as 1 and the sum is 0 plus 1. So, sum as the so gets the a value 1. Now this statement prints this first line, now it will go to the next iteration. Now it checks whether the sum is less than final sum; the sum is 1 which is less than the final sum 15 it will go for the next iteration, it will update

the value of loop variable and the value of sum variable. At the fifth iteration what it does is you have a sum variable 15 which is equal to 15, but not less than 15. This statement is false and it will come out of the loop. So, in this lecture we have seen how the if else family of constructs can be coded in R and how to code for loops and while loops in R.

In the next lecture we are going to see how to perform basic graphics operations in R.

Thank you.