# Foundations of R Software

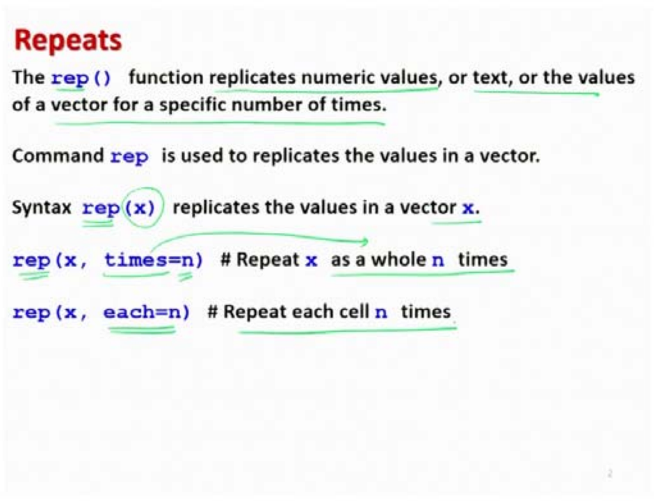
**Prof. Shalabh**
**Department of Mathematics and Statistics**
**Indian Institute of Technology, Kanpur**

## Lecture - 28
## Repeats

Hello, friends. Welcome to the course Foundations of R Software and now, in this lecture we are going to begin with a new topic that is also related to data manipulation, this is about Repeats. So, as the meaning of the name suggest repeat; that means, you want to repeat something you want to do something again and again. So, now, we are trying to learn here that how are we going to repeat some numbers or a sequence of numbers.

Well, these are very small operations, but they are always required when you are trying to do the big programming and you are trying to write down some longer program having some complicated structure, then these types of thing they help you a lot. So, let us try to understand what is repeat, how it works and I will try to once again explain you through various examples so that you can understand them easily. So, let us begin our lecture.

(Refer Slide Time: 01:18)



**Repeats**

The `rep()` function replicates numeric values, or text, or the values of a vector for a specific number of times.

Command `rep` is used to replicates the values in a vector.

Syntax `rep(x)` replicates the values in a vector `x`.

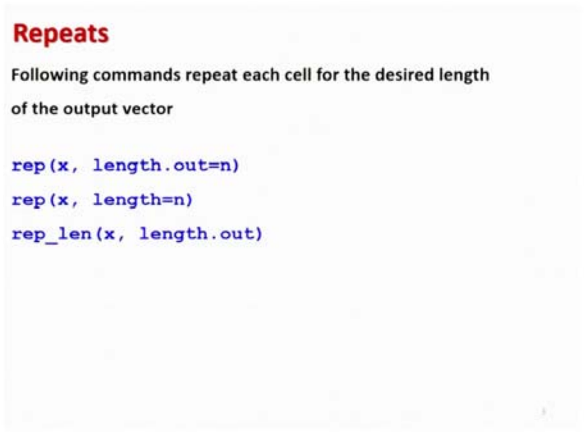
```
rep(x, times=n)   # Repeat x as a whole n times
rep(x, each=n)    # Repeat each cell n times
```

So, we have here a function rep, rep – this is the short form of the repeat. So this rep function replicates the numeric value or takes or the values of a vector for a specific number of times, right. Suppose you want to repeat a value 2 five times or you want to repeat a sequence say three times and so on. So, whatever you want to repeat the syntax is that you try to write down here r e p rep and within the parenthesis you try to write down the value or a vector that you want to repeat.

So, for example, suppose if there is a vector x and you want to repeat so, just write repeat inside the parenthesis x. Now, you have two options whether you want to repeat the entire sequence or you want to repeat the values in the sequence. So, based on that we have here two options with the rep command you can use here option t i m e s times equal to say n or it can be each is equal to n.

So, if you are trying to use this command here times then it is going to repeat the whole data vector n times. And, if you use the command here each is equal to n then it will try to repeat each cell n times.

(Refer Slide Time: 02:45)



## Repeats

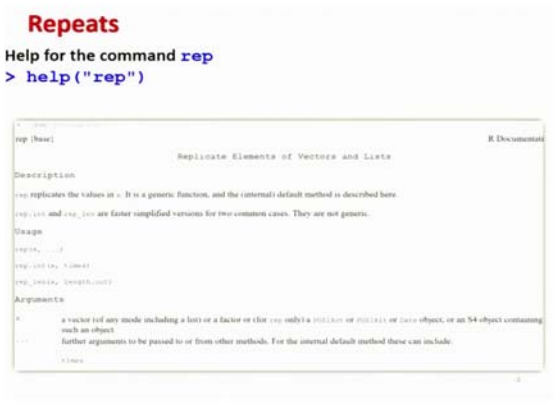
Following commands repeat each cell for the desired length of the output vector

```
rep(x, length.out=n)
rep(x, length=n)
rep_len(x, length.out)
```

So, what it means, how it works let me try to take here some example and then I try to show you. And, before I move further let me try to inform you that we have here three possible options, say, one is here rep inside the parenthesis you write x comma length dot out is equal to n. So, that will give you the output of a desired length say n or you write rep inside the parenthesis x, and then you write here l e n g t h length is equal to small n.

Or you try to write rep underscore l e n and then inside parenthesis you try to write x comma length dot out, then they are going to give you the same output. So, they are going to repeat itself for the desired length, right.

(Refer Slide Time: 03:26)



## Repeats

Help for the command rep

```
> help("rep")
```

rep {base} R Documentation

Replicate Elements of Vectors and Lists

Description

rep replicates the values in x. It is a generic function, and the (internal) default method is described here.

rep.int and rep_len are faster simplified versions for two common cases. They are not generic.

Usage

```
rep(x, ...)
rep.int(x, times)
rep_len(x, length.out)
```

Arguments

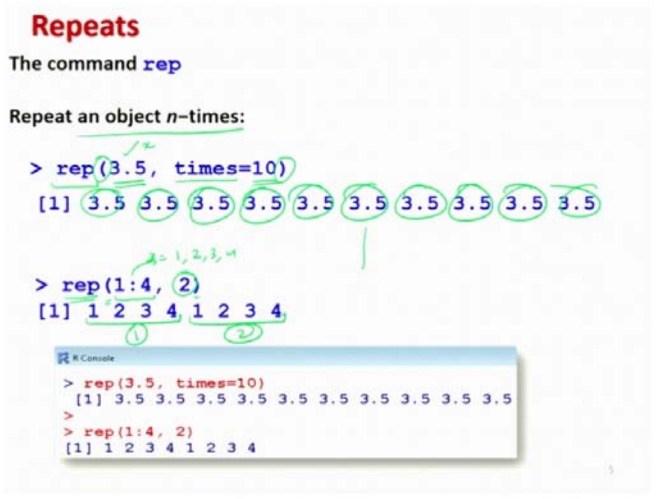
x a vector (of any mode including a list) or a factor or (for rep only) a POSIXct or POSIXlt or Date object, or an S4 object containing such an object.

... further arguments to be passed to or from other methods. For the internal default method these can include:

times

So, in case if you want to learn about this rep command mode, then my suggestion is that please try to look into the help and you can write down here say help inside the parenthesis you write within double quotes you write rep and so on. So, you will get here many many options and all the details about this rep command because I will be choosing here some selected options and then I will try to give you those values which are commonly used.

(Refer Slide Time: 03:43)



## Repeats

The command rep

Repeat an object *n*–times:

```
> rep(3.5, times=10)
 [1] 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5

> rep(1:4, 2)
[1] 1 2 3 4 1 2 3 4
```

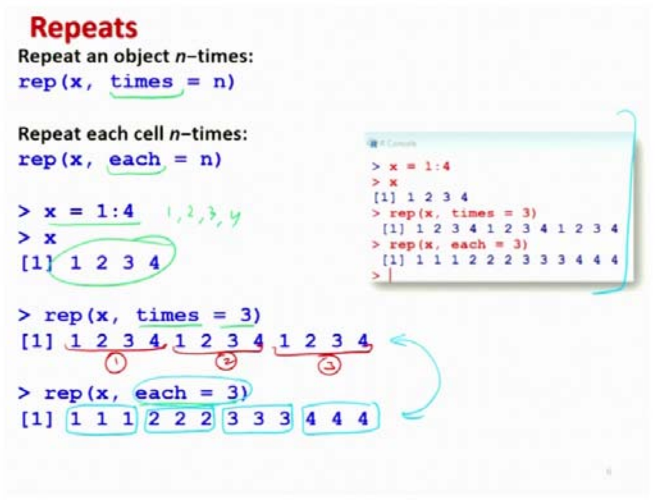
R Console

```
> rep(3.5, times=10)
 [1] 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5
>
> rep(1:4, 2)
[1] 1 2 3 4 1 2 3 4
```

So, first let me try to show you here how you can use the rep command to repeat an object a small n number of times. Suppose, I want to repeat a value 3.5 ten times, I will write down here rep and within parenthesis I will write the number 3.5 which I want to repeat which is over here x and then I will use here the command t i m e s times equal to 10. So, you can see here this will repeat the 3.5 value ten times 1 2 3 4 5 6 7 8 9 10, right.

And, similarly in case if you write down here for example, rep and then within parenthesis you write 1 colon 4, so, what is the meaning of this 1 colon 4? Now, you know this is 1 2 3 and 4 this is your here x, this is a vector and then you are trying to write down here 2. So, then it is going to repeat it 2 times. You can see here 1 2 3 4 – 1 time and 1 2 3 4 2 times that is 2. So, you can see here that this command rep can be used for a scalar like 3.5 and a vector like here 1 2 3 4 and this is here the screenshot.

(Refer Slide Time: 05:01)



## Repeats

Repeat an object *n*–times:

```
rep(x, times = n)
```

Repeat each cell *n*–times:

```
rep(x, each = n)
```

```
> x = 1:4
> x
[1] 1 2 3 4

> rep(x, times = 3)
[1] 1 2 3 4 1 2 3 4 1 2 3 4

> rep(x, each = 3)
[1] 1 1 1 2 2 2 3 3 3 4 4 4
```

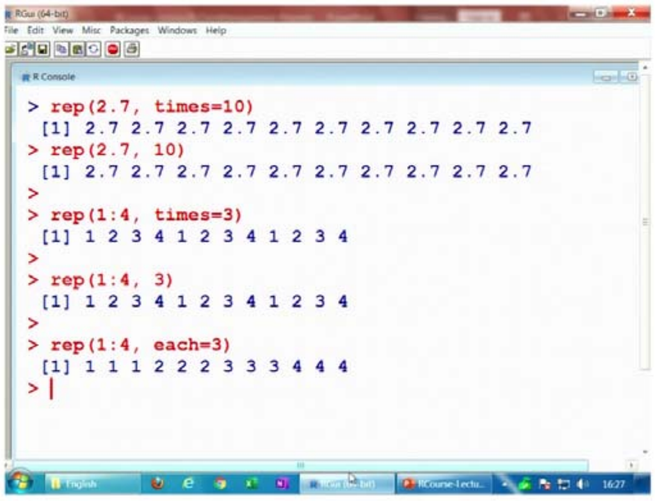
```
> x = 1:4
> x
[1] 1 2 3 4
> rep(x, times = 3)
 [1] 1 2 3 4 1 2 3 4 1 2 3 4
> rep(x, each = 3)
 [1] 1 1 1 2 2 2 3 3 3 4 4 4
>
```

So, now after this I try to illustrate here that what will be the difference in the outcome when we are trying to use the option times t i m e s and each e a c h, right. So, I will try to use it on the same input and then I will try to explain you here how their outcomes will look. So, if I try to take here data vector say 1 2 4 it has 4 values 1 2 3 and 4 like this. So, now, I try to repeat this x here using the option t i m e s is equal to 3 and e a c h is equal to 3. Now, you see how the outcome looks like.

Well, I am trying to use here the option times you can see here this is the repetition 1 time, 2 times and 3 times. So, the entire sequence is repeated 3 times. Now, I try to use here the option each like as here like this each is equal to 3. So, now, what will happen? The first element or the element in the first cell of the data vector which is here 1, this is going to be repeated here 3 times. 1 is repeated 3 times, 2 is repeated 3 times, 3 is repeated 3 times and 4 is repeated 4 times.

So, you can see here this is how their outcomes will differ and this is here the screenshot of the same operation. So, let me try to show you first these things on the R console and then I will try to show you something more.

(Refer Slide Time: 06:27)



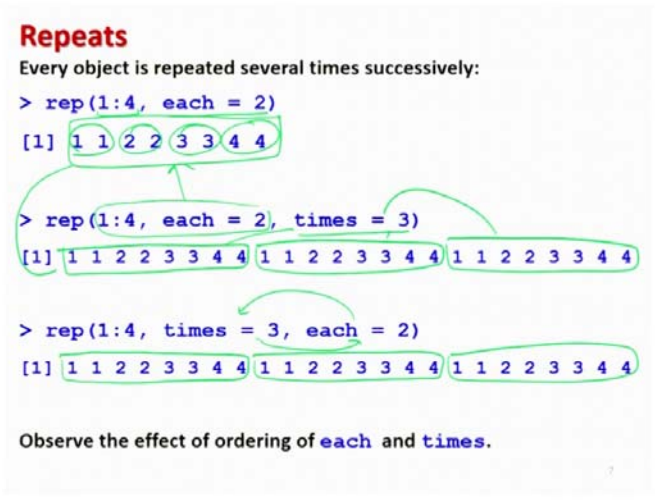
```
> rep(2.7, times=10)
 [1] 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7
> rep(2.7, 10)
 [1] 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7 2.7
>
> rep(1:4, times=3)
 [1] 1 2 3 4 1 2 3 4 1 2 3 4
>
> rep(1:4, 3)
 [1] 1 2 3 4 1 2 3 4 1 2 3 4
>
> rep(1:4, each=3)
 [1] 1 1 1 2 2 2 3 3 3 4 4 4
> |
```

So, if I try to say here rep say 2.7 say times equal to suppose here 10 you can see here this is repeated 10 times and even if you do not give here this option times or you do not write here time, but if you simply write down here 10, then it is going to be here like this. Now, in case if you try to write down here say here we try to repeat a sequence say 1 to 4 and times is equal to say 3. So, you can see here 1 2 3 4, 1 2 3 4 and 1 2 3 4 they are repeated 3 times.

And, now the same thing if you try to do here that if you do not write here anything and you try to simply say here rep 1 to 4 and 3 you get the same thing here. And, if you try to use here the command here each in place of times so, what will happen here? Each is equal to 3 you can see here 1 1 1 2 2 2 3 3 3 and 4 4 4 all the numbers are repeated 3 times.

So, one thing you can see here that when you are writing here times and you are not writing anything, but you are simply writing the numbers, so, whatever is the value at the second place after the comma that is always going to be considered as t i m e s times by the R software. That is the default choice. So, unless and until you write here each it is not going to be considered as each. So, that is what you have to keep in mind when you are trying to execute these operations, right.

(Refer Slide Time: 07:55)



## Repeats

Every object is repeated several times successively:

```
> rep(1:4, each = 2)
[1] 1 1 2 2 3 3 4 4

> rep(1:4, each = 2, times = 3)
[1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4

> rep(1:4, times = 3, each = 2)
[1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4
```

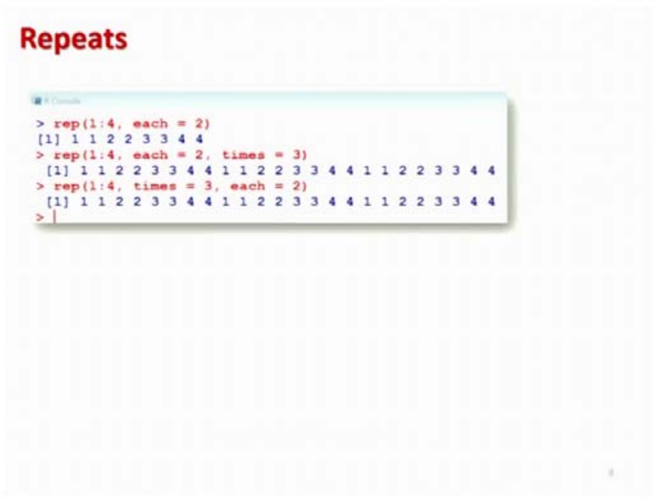
Observe the effect of ordering of each and times.

Now, I try to show you here the outcome that what happens when you are trying to use each and times command both together. So, if you try to see here I take here a sequence here 1 to 4, 1 2 3 4 and I say here each is equal to 2. So, you know means every value 1 2 3 and 4 they are going to be repeated 2 times. Now, in this case I try to add here the option times equal to 3.

So, now, what will happen here? Now, you can see. The first this will be executed and the outcome of this is given here like this and if you try to see the same thing is repeated here again 1 2 3 4 like as a like and this is repeated 3 times, right. On the other hand, in case if you try to write down the same command in this format that you try to interchange the positions of this times in each that will not make any change.

So, you can see here 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 and 1 1 2 2 3 3 4 4 they are the same, right. There is no change. So, that is what you have to observe and keep in mind that how R works right, ok.

(Refer Slide Time: 09:04)

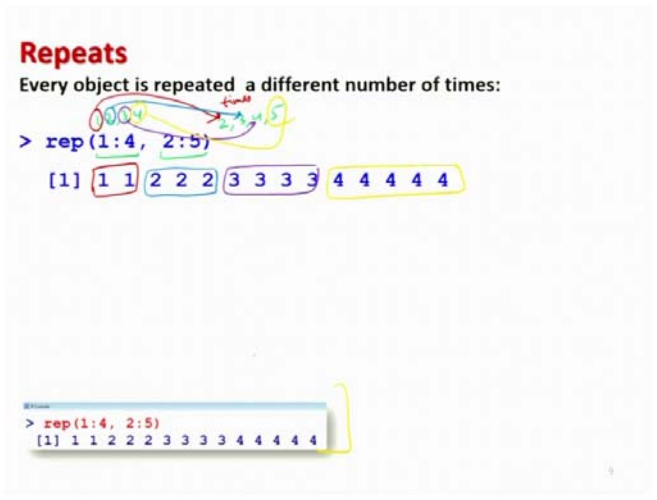


## Repeats

```
> rep(1:4, each = 2)
[1] 1 1 2 2 3 3 4 4
> rep(1:4, each = 2, times = 3)
 [1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4
> rep(1:4, times = 3, each = 2)
 [1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4
>
```

And, this is here the screenshot of the same operation you can see.

(Refer Slide Time: 09:07)



## Repeats

Every object is repeated a different number of times:

```
> rep(1:4, 2:5)
  [1] 1 1 2 2 2 3 3 3 3 4 4 4 4 4
```

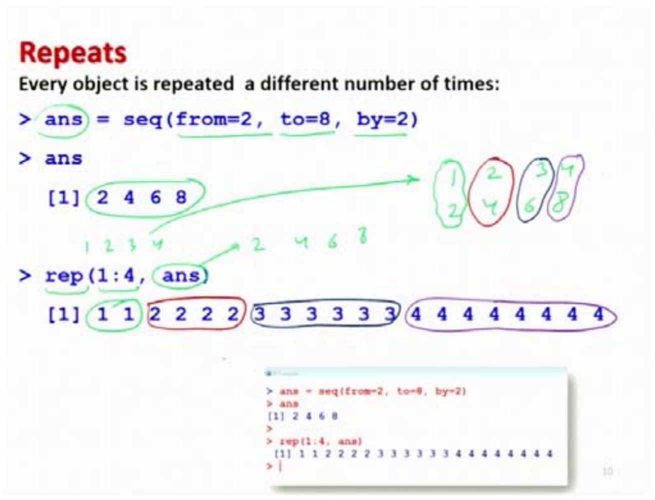
```
> rep(1:4, 2:5)
 [1] 1 1 2 2 2 3 3 3 3 4 4 4 4 4
```

Now, I give you here one more example that suppose I try to take here a sequence like as 1 to 4; that means, there are four values 1 2 3 and 4, and after that I write down here another sequence 2 to 5 that is 2 3 4 and 5, right. So, now, what is going to happen? So, first you have to observe the way it is working. So, first of all this is going to be taken as say here times. So, first of all this value 1 is picked up and this is repeated 2 times you can see here.

Then after this the next value 2 is picked up and this is repeated 3 times you can see here. Then after this the third value 3 is picked up and it is repeated 4 times you can see here and finally, the last value 4 is picked up and it is repeated 5 times you can see here. So, this is another way the repeat command works when both are the data vector. So, this is what you have to understand that how these operations are going to work, right.

And, this is here in the screenshot so, you can be confident that ok that when are you going to do it on the R console the same outcome will be there, ok.

(Refer Slide Time: 10:19)



## Repeats

Every object is repeated a different number of times:

```
> ans = seq(from=2, to=8, by=2)
> ans
  [1] 2 4 6 8
> rep(1:4, ans)
  [1] 1 1 2 2 2 2 3 3 3 3 3 3 4 4 4 4 4 4 4 4
```

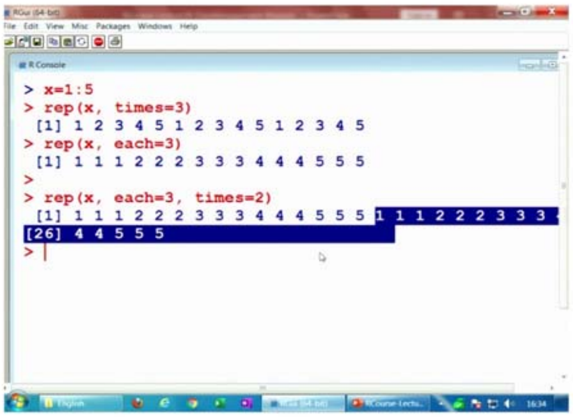
```
> ans = seq(from=2, to=8, by=2)
> ans
[1] 2 4 6 8
>
> rep(1:4, ans)
 [1] 1 1 2 2 2 2 3 3 3 3 3 3 4 4 4 4 4 4 4 4
> |
```

Now, I try to show you here the similar operation where every object is repeated for a different number of times, right. So, suppose I try to create here a sequence from 2 up to here 8 and by 2. So, this sequence is going to be like 2 4 6 8 and it is value is stored in the variable a n s answer. And, now you try to write down here command here rep this 1 colon 4 that is 1 2 3 and 4 and what a n s times. So, this is here 2 4 6 and here 8.

So, what will happen? The same thing will happen that let me try to write down here like this 1 2 3 and here 4 and then 2 4 6 8. So, 1 is going to be repeated for 2 times you can see here, then next 2 is going to be repeated 4 times, then it will pick up the third value 3 is going to be repeated 6 times. So, 1 2 3 4 5 6 and finally, the last value. What is the last value? 4. So, this is 4 is going to be repeated 8 times. So, you can see here 1 2 3 4 and then 1 2 3 4 that is 8 times.

So, what you have to observe that how this R is going to work with these commands that will help you in creating more such sequences using the repetitions, right.

(Refer Slide Time: 11:54)

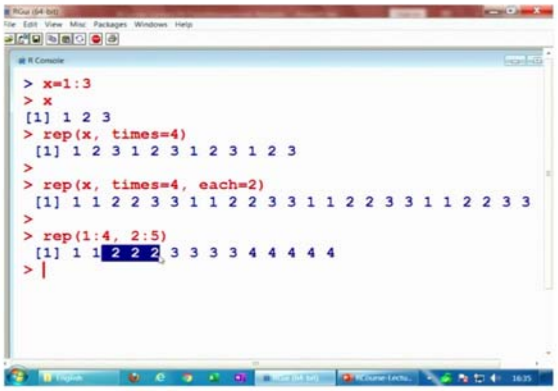


```
> x=1:5
> rep(x, times=3)
 [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
> rep(x, each=3)
 [1] 1 1 1 2 2 2 3 3 3 4 4 4 5 5 5
>
> rep(x, each=3, times=2)
 [1] 1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 1 1 1 2 2 2 3 3 3
[26] 4 4 5 5 5
> |
```

And now before going further let me try to show you these things on the R console also so that you get confidence that ok that these things are working, right. So, let me try to take here the data vector here as say 1 to suppose 5 I take and then I try to say here repeat x a times equal to 3 and then I try to write down here x try to repeat for each equal to 3. So, you can see here this is happening like this, one more example, not bad right.

And, then I try to take care it like this that I try to take care suppose repeat each equal to 3 and then also I try to add here post 2. So, you can see here that in this case this 1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 this is one time and this is here 1 1 1 2 2 2 3 3 3 and like this is second time. So, that is going to happen.

(Refer Slide Time: 12:36)

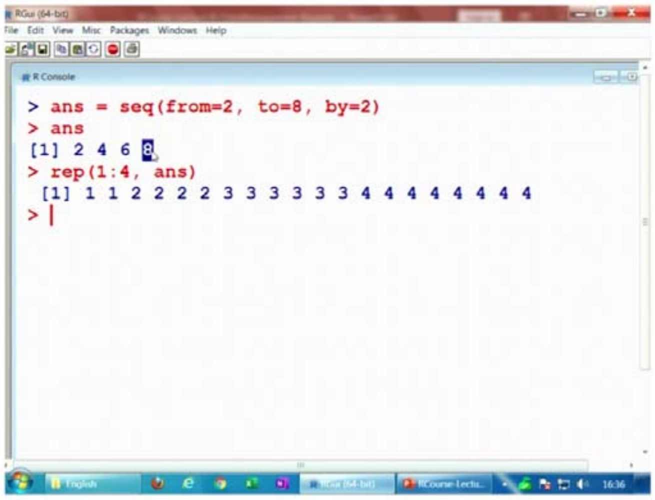


```
> x=1:3
> x
[1] 1 2 3
> rep(x, times=4)
 [1] 1 2 3 1 2 3 1 2 3 1 2 3
>
> rep(x, times=4, each=2)
 [1] 1 1 2 2 3 3 1 1 2 2 3 3 1 1 2 2 3 3 1 1 2 2 3 3
>
> rep(1:4, 2:5)
 [1] 1 1 2 2 2 3 3 3 3 4 4 4 4 4
> |
```

Now, in case if I try to take here one more example suppose x takes the value here 1 to suppose 3, x is here like this. And if you try to write down here rep x times equal to suppose here 4 it will give you like this 1 2 3 is going to be repeated 4 times you can see 1 2 3 and here 4. And, now try to see here in the same command try to use here each is equal to suppose 2 times. So, you can see here now each is going to overtake and now 1 2 3 is each of the cell is repeated 2 times and then it is repeated 4 times.

So, this is what you have to keep in mind when you are trying to work in the R software that how actually R works. So, whosoever has done the programming I means you have to follow that rule unless until you try to write your own program. Similarly, if you try to see here what will happen to here rep 1 to 4, 2 to 5 as I shown you? This is the same command that 1 is repeated 2 time, 2 is and then 2 is repeated 3 times and so on, right. So, that is here like this.

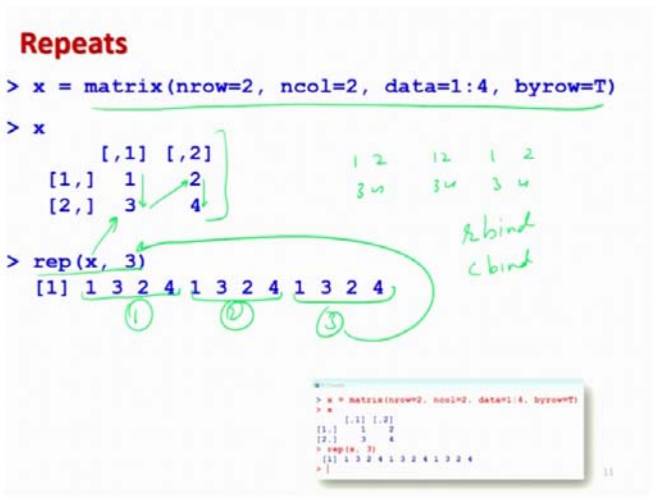
(Refer Slide Time: 13:43)



```
> ans = seq(from=2, to=8, by=2)
> ans
[1] 2 4 6 8
> rep(1:4, ans)
 [1] 1 1 2 2 2 2 3 3 3 3 3 3 4 4 4 4 4 4 4 4
> 
```

And, in case if you try to say here like this the same example which I shown you here on this one if you try to write down here your answer is going to be like this a n s and then after that if you try to repeat this value what will happen here? You try to repeat the sequence 1 to 4 for the times which is controlled by this a n s values. So, 1 is going to be controlled or repeated 2 times, 2 is going to be repeated 4 times, 3 is going to be repeated 6 times and 4 is going to be repeated 8 times.

(Refer Slide Time: 14:02)



## Repeats

```
> x = matrix(nrow=2, ncol=2, data=1:4, byrow=T)
> x
     [,1] [,2]
[1,]    1    2
[2,]    3    4
> rep(x, 3)
 [1] 1 3 2 4 1 3 2 4 1 3 2 4
```

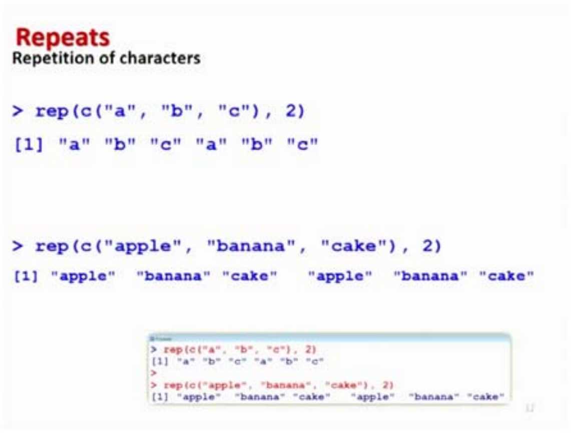
So, you can see here these are not very difficult operations, but you have to understand how they are trying to give you the output. Now, I try to give you take care one more example where I am trying to consider here a matrix of order 2 by 2 in which the data is 1 2 3 4 and the observations are arranged by rows. You already have learnt the matrix operation, so, x matrix here look like this.

Now, in case if you try to use here the command rep x, 3 then you know the way it is going to work it is very different. So, you have to understand, right. So, if you try to see here the outcome here is 1 3 2 4 1 3 2 4 and 1 3 2 4 this is 1 times, 2 times and 3 times. So, this 3 is corresponding to this 3 and now this here x here is like this, but the data has been arranged like 1 3 2 and here 4.

So, you have to understand how the matrix is going to be repeated when you use the rep command well actually you do not want this thing you wanted that this matrix x is like this 1 2 3 4 this and then 1 2 3 1 2 here 3 4 like this and 1 2 3 4 like this. So, for that if you remember you had the command like bind it was rbind or says cbind you have done. These two command for binding row wise or column wise.

So, just be careful that was my objective to show you that when you are trying to operate it with the matrix what is going to happen, right, ok.

(Refer Slide Time: 15:24)



## Repeats
Repetition of characters

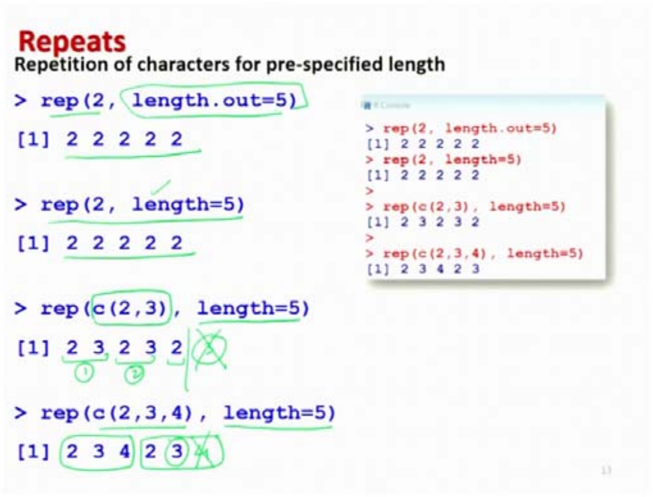
```
> rep(c("a", "b", "c"), 2)
[1] "a" "b" "c" "a" "b" "c"


> rep(c("apple", "banana", "cake"), 2)
[1] "apple"  "banana" "cake"   "apple"  "banana" "cake"
```

```
> rep(c("a", "b", "c"), 2)
[1] "a" "b" "c" "a" "b" "c"
>
> rep(c("apple", "banana", "cake"), 2)
[1] "apple"  "banana" "cake"   "apple"  "banana" "cake"
```

So, now after this whatever repetition command you have learnt for the numbers they are also valid for character. So, if you try to see here I try to take here a data vector consisting of three alphabets which are given inside the double quotes a, b and c and I try to repeat them two times that is times equal to 2. So, you can see here that this a b c and a b c they are repeated two times because here the times is the default.

Now, similarly if you try to take here instead of a b c if you try to take here apple, banana, cake and you try to repeat it 2 times we can see here apple, banana, cake and apple, banana, cake both are repeated here 2 times, ok.

(Refer Slide Time: 16:01)



## Repeats
Repetition of characters for pre-specified length

```
> rep(2, length.out=5)
[1] 2 2 2 2 2

> rep(2, length=5)
[1] 2 2 2 2 2

> rep(c(2,3), length=5)
[1] 2 3 2 3 2

> rep(c(2,3,4), length=5)
[1] 2 3 4 2 3
```

```
> rep(2, length.out=5)
[1] 2 2 2 2 2
> rep(2, length=5)
[1] 2 2 2 2 2
>
> rep(c(2,3), length=5)
[1] 2 3 2 3 2
>
> rep(c(2,3,4), length=5)
[1] 2 3 4 2 3
```

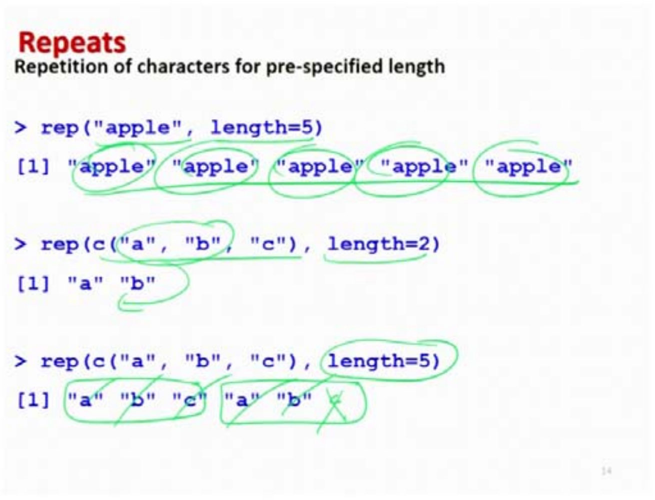
And, similarly if you want to have here some more operations like this one that I want to repeat here to, but how many times? The length of the data vector should be 5; that means, five times. So, my length is pre specified. Now, in a univariate case you may not see the difference, but as soon as I try to take data vector then you will see the difference that how the outcome changes.

So, if you try to see here 2 and I try to give here the option length dot out is equal to 5, it will give you here five values, no issues. And, this is the same thing if you try to use here rep 2 and this option length is equal to 5 it will again give you the five values no issue, but if you try to take here a data vector then you try to see what happens. If you try to take a data vector of two values 2 and 3 and if you give here an option length is equal to 5, so, that means, you need only five values.

So, this 2 3 will start repeating 1 time there are two values; 2 times there are two values. So, now, there is a space only for one value. So, this ideally it should be here 3, but this 3 will not come here and process will start will stop only at 2, right. And, similarly if you try to take here three elements 2, 3 and 4 and still you try to take the length is equal to 5.

So, this 2, 3, 4 is going to be repeated here and then once again 2, 3, 4 will be repeated, but then there is no place for the last value 4 because the fifth place is occupied by 3. So, that is why it will stop here and you will get here only 2 3 4 and 2 3, right. So, this is what you have to keep in mind.

(Refer Slide Time: 17:33)



## Repeats

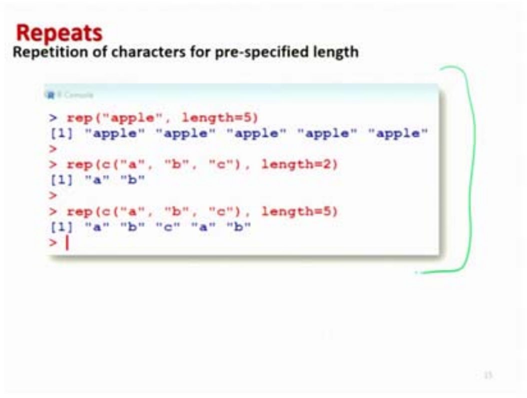
Repetition of characters for pre-specified length

```
> rep("apple", length=5)
[1] "apple" "apple" "apple" "apple" "apple"

> rep(c("a", "b", "c"), length=2)
[1] "a" "b"

> rep(c("a", "b", "c"), length=5)
[1] "a" "b" "c" "a" "b"
```

And, the same thing will also happen with the character that if you try to take here the character say apple and you try to repeat it 5 times. So, you will get here apple 5 times you can see here 1 2 3 4 and 5 and similarly, if you try to take here a sequence of a, b, c, but your length here is 2. So, only the first two letters a and b will be printed here, right.

And, if you try to take here length is equal to 5 then a, b, c will be repeated here and ideally after this a, b, c has to be repeated again, but there is no place for the c because your limit is that you want only 5 characters. So, 1, 2, 3, 4 and 5 the process will stop at b and no c will enter here.

(Refer Slide Time: 18:19)



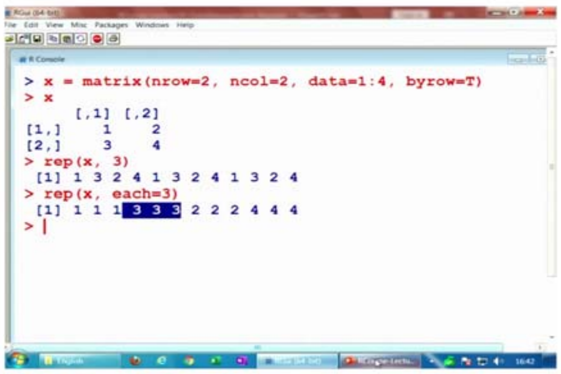
## Repeats

Repetition of characters for pre-specified length

```
> rep("apple", length=5)
[1] "apple" "apple" "apple" "apple" "apple"
>
> rep(c("a", "b", "c"), length=2)
[1] "a" "b"
>
> rep(c("a", "b", "c"), length=5)
[1] "a" "b" "c" "a" "b"
>
```

So, that is how the things will work and this is here the screenshot and let me try to show you these things on the R console how do they work and now you have seen that they are not very difficult thing to do, right.

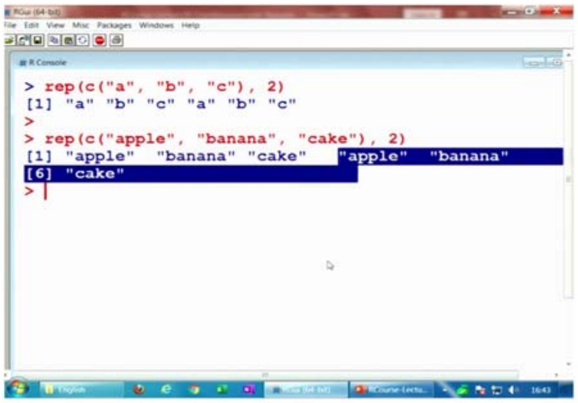
(Refer Slide Time: 18:30)



```
> x = matrix(nrow=2, ncol=2, data=1:4, byrow=T)
> x
     [,1] [,2]
[1,]    1    2
[2,]    3    4
> rep(x, 3)
 [1] 1 3 2 4 1 3 2 4 1 3 2 4
> rep(x, each=3)
 [1] 1 1 1 3 3 3 2 2 2 4 4 4
>
```

So, if you try to see here first look let me try to take this matrix example. So, you can see here this is your here matrix and then you try to repeat this matrix three times. So, you can see here like this, right. In this case if you try to write down here each is equal to 3 then you can see this outcome is different that each of this value in the same order 1 3 2 4 this is repeated three times 1, then 3, then 2 and then 4, right.

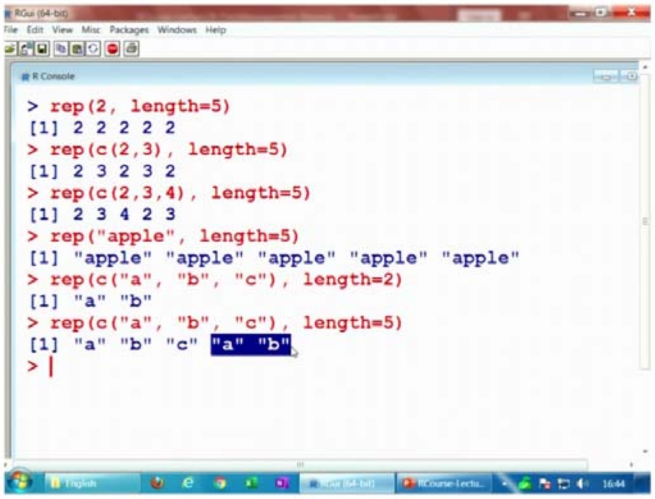
(Refer Slide Time: 18:55)



```
> rep(c("a", "b", "c"), 2)
[1] "a" "b" "c" "a" "b" "c"
>
> rep(c("apple", "banana", "cake"), 2)
[1] "apple"  "banana" "cake"   "apple"  "banana"
[6] "cake"
> 
```

And, similarly if you try to take here some here character and you want to repeat them so, it is here like this the sequence of a, b, c is repeated here 2 times. Similarly, the sequence of apple, banana and cake and if you want to repeat it 2 times you can see here apple, banana, cake and apple, banana, cake are repeated 2 times.

(Refer Slide Time: 19:19)



```
> rep(2, length=5)
[1] 2 2 2 2 2
> rep(c(2,3), length=5)
[1] 2 3 2 3 2
> rep(c(2,3,4), length=5)
[1] 2 3 4 2 3
> rep("apple", length=5)
[1] "apple" "apple" "apple" "apple" "apple"
> rep(c("a", "b", "c"), length=2)
[1] "a" "b"
> rep(c("a", "b", "c"), length=5)
[1] "a" "b" "c" "a" "b"
> 
```

Similarly, if you want to have here these type of commands so, you can see here that if you try to use this command here. If you are trying to repeat the value 2 for five times it will be repeated five times, but in case if you try to take here a data vectors say here 2 comma 3 and if you try to repeat it 5 times you can see here the answer comes out to be 2, 3, 2, 3 and after that it stops at 2, right.

And, if you try to increase this number suppose if I say 2, 3, 4 then you can see here the whole vector 2 3 4 will come here, but after that after 2 and 3 it will stop because there is a limit that you cannot increase this sequence beyond five elements, right. So, similarly if you try to see here for this sequence of characters also this is valid that you are trying to repeat the apple for five times and if you try to repeat here this 3 values a b c two times.

So, you can see here it is only here a and b are appearing here, but if you try to make it here suppose here 5. So, you can see here a b c will come here and after that only a b will come here, right. So, this is how we try to do all such calculations without any problem, ok. So, now we come to an end to this lecture and you can see here we have done very simple operation today.

We have learnt how we can generate the sequence in which the values are getting repeated and yes, these things are going to be useful when you are trying to do the programming, you are trying to handle the databases, you are trying to generate different types of reports at that place the importance of these basic commands will be seen.

At this moment you just try to learn it as such that if you are asked to repeat the values how are you going to do it, right. So, now it is you it is your turn that you try to take couple of example based on numbers, based on character, try to use each times etcetera their combination and try to see how this is working.

You have seen that there is a special way in which the R is trying to repeat the values when you try to take each or times or a combination of them or you try to repeat them with the sequences means both the times and the data vector both are in the form of data vector that you want to repeat. So, once you try to understand and the main thing is not to get the output, but try to understand that how this is working.

So, you try to practice it and I will see you in the next lecture. Till then, goodbye.