

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

Strings - Display and Formatting
Lecture - 36
Print and Format with Concatenate

Hello, friend. Welcome to the course Foundations of R Software. You can recall that in the last lecture we started a discussion on the print command and we wanted to learn how we can generate a report in which there are many types of formatting is also needed. So, we discussed the command Print and we had used the format option.

And, now today I will try to continue on the same lines and I will try to give you here some more options like a cat. So, I will try to discuss that if you want to do something then if you try to use the print command, then what happens and if you try to use cat command then what happens and with the format how you can control the output. So, let us try to begin this lecture and try to understand these concept through some examples, ok.

(Refer Slide Time: 01:06)

Formatting and Display of Strings

Suppose we want to print

The zero occurs at 2*pi radians.

Character

$\pi = \pi = \frac{22}{7} = 3.14 \dots$ $2 \times 3.14 = 6.28$

The zero occurs at 6.18 radians

mathematical.

Suppose you want to print the zero occurs at twice of pi radian; pi you know pi is the mathematical pi whose value is 22 by 7 like as 3.14 etc. So, pi is the word or a command

in R that gives you the value of pi. So, I want that let R takes this value here pi itself and it give me the value like the zero occurs at say 2 into 3.14 is something like 6.18 at 6.18 radian this is what I want to do, right.

(Refer Slide Time: 02:07)

Formatting and Display of Strings

The `print` function has a significant limitation that it prints only one object at a time.

Trying to print multiple items gives error message:

```
> print("The zero occurs at", 2*pi, "radians.")
Error in print.default("The zero occurs at", 2 *
pi, "radians.") : invalid 'quote' argument
```

```
> print("The zero occurs at", 2*pi, "radians.")
Error in print.default("The zero occurs at", 2 * pi, "radians.") :
invalid 'quote' argument
```

Now, what I try to do here I try to use here the print command and see. So, suppose I can also use here my this R knowledge and what I can do here that I can divide this sentence into two parts – first part which is character and another is here mathematical and then here character.

You can see here this first and third part they are here the character and this twice of pi is mathematical, right. So, now, let us try to use this command here and because I know from my past knowledge that I have to give the characters inside the double quotes and the mathematical operation just like that. So, what I try to do here that I try to write down here the zero occurs at inside the double quotes, then I try to write down here twice star pi and they are separated by comma and then I try to write down here radian inside the double quotes.

But, as soon as you use the command print over this value, it gives you here an error, right. So, why this is happening and why print is not working and what is the way out? That is the question which we would like to explore in this lecture. So, the print function

has a significant limitation that it prints only one object at a time and here what are you trying to do?

You have here a character and you have here a numeric and you want to print both of them together. So, the question is now how to do it? Ok, first option is that ok means I can bring this print command at various places wherever we have character and numeric. So, I can write down here print for this zero occur set.

(Refer Slide Time: 03:17)

Formatting and Display of Strings

The only way to print multiple items is to print them one at a time

```
> print("The zero occurs at"); print(2*pi);  
print("radians")
```

```
[1] "The zero occurs at"  
[1] 6.283185  
[1] "radians"
```

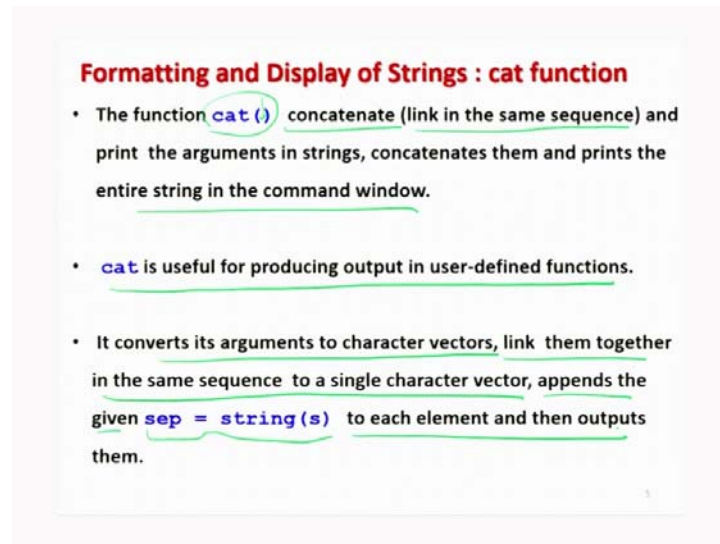
The `cat` function is an alternative to print that lets you combine multiple items into a continuous output.

And, then I can write down here say here print twice star pi and then I can write down here print radiance. And, let us try to see what happens like this print the zero occurs at and then yeah, I am giving here this semicolon and then I am writing here print 2 star pi then I am writing here once again semicolon and then I am writing down here print radiance. So, if you try to recall what you wanted, you wanted like this the zero occurs at 6.18 radians, but now it is giving you here like this the zero occurs at 6.283185 and then radians.

So, is it the same format in which you wanted to have it? No. This outcome is like that you are going to get this outcome in three different lines – line number 1, line number 2 and line number 3 and if you try to do it in the R console it will look like this is the thing which you do not want you want everything to be to be written in the same line. So, how to get it done?

So, in order to solve such issues we have here an alternative function which is here `cat` and this is an alternative to `print` and `print` function and it allows you to combine the multiple items into a continuous output. For example, here you wanted to print the character number and then character. So, these are the things which can be done with the `cat` function.

(Refer Slide Time: 04:28)



Formatting and Display of Strings : cat function

- The function `cat()` concatenate (link in the same sequence) and print the arguments in strings, concatenates them and prints the entire string in the command window.
- `cat` is useful for producing output in user-defined functions.
- It converts its arguments to character vectors, link them together in the same sequence to a single character vector, appends the given `sep = string(s)` to each element and then outputs them.

So, the first question comes here. What is the meaning of this `cat`? `Cat` is not a `cat`, right which speak like `meow meow`, right. `Cat` is a short form of `concatenate`. What is the meaning of this `concatenate`? That means, it is trying to link in the same sequence. So, this function `cat` this links different numbers and characters in the same sequence what you try to give it inside the parenthesis, inside the arguments and then it prints the entire string in the command window, right.

Remember, why I am emphasizing on the command window or printing in the command window because if somebody wants to print it on a printer, then this print is not really going to work. `Cat` is not really going to work print and command they are used only to print on the screen on the command window, right.

And, this `cat` is quite useful when we want to produce the output in a user defined function in a user friendly way, right? Actually what it does? It converts its argument to character vector then links them together in the same sequence to a single character

vector. And, after that in case if you try to use here an option like sep then it will try to append the given separator which is obtained through sep to each of the element and then outputs it.

(Refer Slide Time: 05:40)

Formatting and Display of Strings : cat function

Usage

```
cat(... )  
cat(... , file = \"\", sep = \" \", fill = FALSE,  
labels = NULL, append = FALSE)
```

cat puts a space between each item by default. >

One must provide a newline character `\n` (newline) to terminate the line.

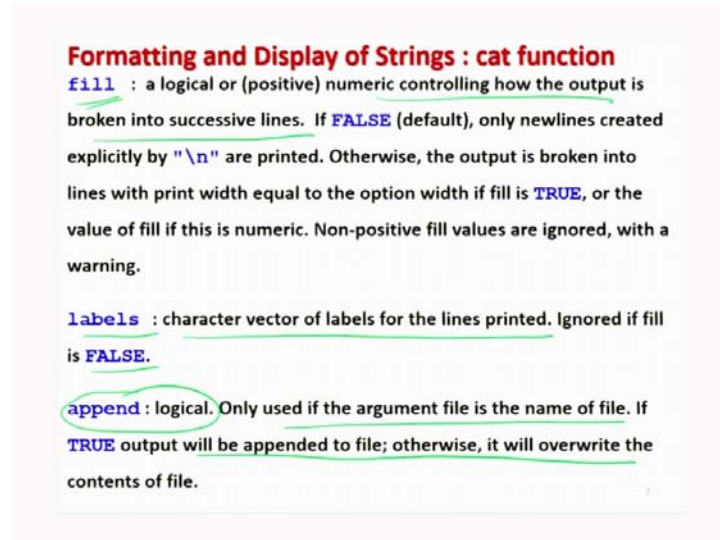
What does this mean? I will try to show you with an example. So, the usual function is here cat and within the parenthesis you have to give that what you want to print and then you have to give various options. So, there are various options here. So, if you want to print something inside a file, then you have to specify here the name of the file in which you want to print the output, then we have here sep which means actually separator.

Separator means it will try to separate the different say strings and numbers whatever you are going to give inside the argument inside the parenthesis and then it has here fill that I will try to show you what it does, then we have a labels append etcetera. So, as the name suggest for example, append; append means whatever you have printed you have to print after this, right.

And, then I can show you here or I can inform you in advance that there is a character here like this backslash n. So, this is an indicator of the new line, right. So, the outcome otherwise you will see once the outcome of the cat comes, then the pointer or the symbol which is here on the R console that will be just after the output, but you want to bring it

on the next line. So, if you want to bring the this prompt symbol to the next line then you have to use this character.

(Refer Slide Time: 06:51)



I will try to show you once again with the one more thing and then we have there are many more things about this cat and I would try to and I would request you that you try to read it from the help menu some important things I have written here, yeah.

For example, this option here is fill, this is a logical variable. It gives the value in terms of true and false and it tries to control how the output is to be broken into the successive line. Similarly, we have here labels and this is a character vector of label for the lines which are printed and it is ignored if fill is FALSE. So, that is also a logical variable having the value TRUE and FALSE.

And, simply append is also logical variable. It takes the value in terms of TRUE and FALSE and it is to be used only if the argument file is the name of the file means when you want to save the outcome into the file and so on. So, if it is TRUE, then the output will be appended to file; otherwise it will be overwrite on the contents of the file.

So, if you have written if you have already written something and if you want the output to be written at the end means every time you try to repeat the your program whatever the outcomes outcome comes if that is to be written after the end then you have to use

TRUE otherwise if you want to overwrite the earlier outcome you can you need to write here FALSE.

(Refer Slide Time: 07:59)

Formatting and Display of Strings: print vs cat

The only way to print multiple items is to print them one at a time

```
> print("The zero occurs at"); print(2*pi);  
print("radians")  
[1] "The zero occurs at"  
[1] 6.283185  
[1] "radians"
```

The `cat` function is an alternative to print that lets you combine multiple items into a continuous output:

```
> cat("The zero occurs at", 2*pi, "radians.",  
"\n")  
The zero occurs at 6.283185 radians.
```

So, there are many such options and I would request you that you please try to look into the help. Now, I try to give you here couple of examples so that I can show you what is the outcome of the cat and first I would like to show you what is the difference in the outcomes of the print and cat.

So, I try to take here the same example which I just took it that I wanted to print the 0 occurs at 2 pi radian. So, once you are trying to use the print command with each and every character or numeric values, then it is going to print the outcome in three different lines, but you want to have it on the single line.

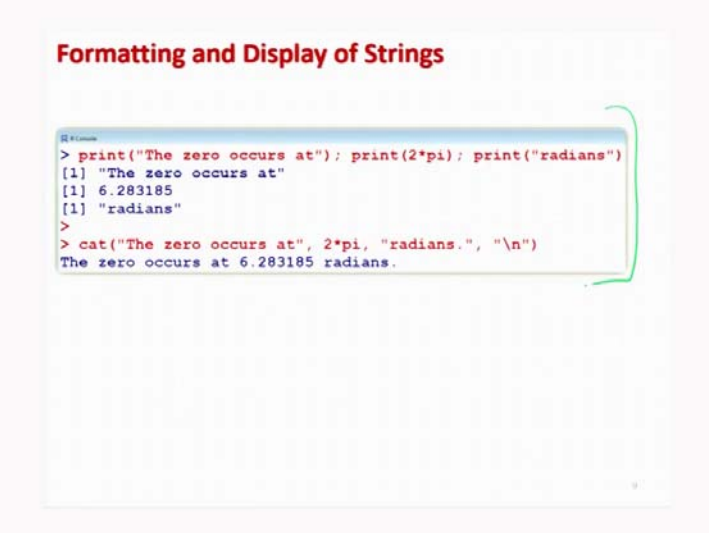
So, then you can use here the cat command and if you try to see here that simply have to write down here cat and then whatever are the characters they have to be written inside the double quotes, whatever are the numerical values they have to be written without using the double quotes and everything is going to be separated by comma. That is a very simple format for writing the commands or what you want to print with the cat commands.

So, for example, if you want to print here the zero occurs at so, you know this is here a character. So, you try to write here like this. Then you know the twice of pi which is 2

into pi that is 2 multiplication pi this is a number. So, I am just typing it here as a 2 into pi and then this radians, radians is a character. So, I am trying to give it inside the double quotes and then all these things they are separated by the comma and now, after this I try to give you here this symbol within double quotes backslash.

And, so, this is going to change the line after you have executed it and the control will come to the next line. So, if you try to execute it on the R software, it will look like this the zero occurs at this 6.283185 radian. So, exactly this is what you wanted. But, when you use the print command it give you here this type of outcome. So, this is what you understand that what is the difference between print and cat. Print and cat both are doing the same thing, but the way in which they are going to operate they are different, right.

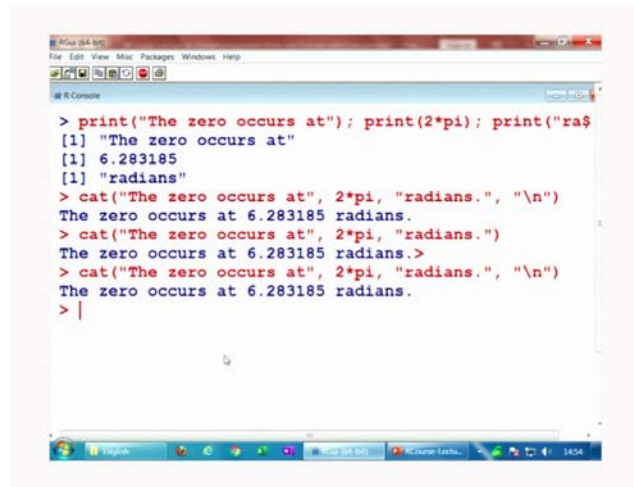
(Refer Slide Time: 09:49)



```
Formatting and Display of Strings
R Console
> print("The zero occurs at"); print(2*pi); print("radians")
[1] "The zero occurs at"
[1] 6.283185
[1] "radians"
>
> cat("The zero occurs at", 2*pi, "radians.", "\n")
The zero occurs at 6.283185 radians.
```

And, if you try to see here this is the screenshot of the same outcome, right. So, let me try to show you these things first on R console and then I try to give you here some more commands over here.

(Refer Slide Time: 10:02)

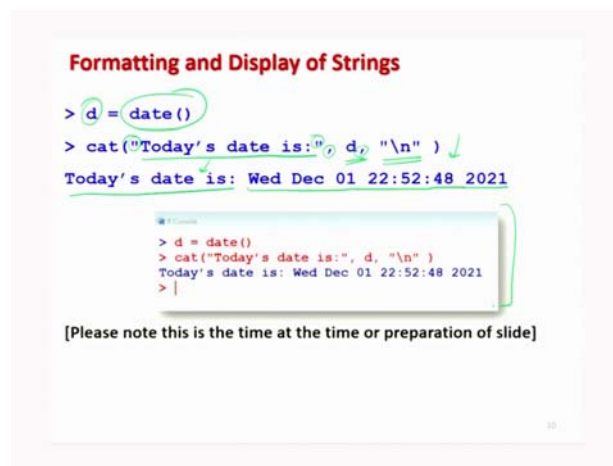


```
> print("The zero occurs at"); print(2*pi); print("ra$
[1] "The zero occurs at"
[1] 6.283185
[1] "radians"
> cat("The zero occurs at", 2*pi, "radians.", "\n")
The zero occurs at 6.283185 radians.
> cat("The zero occurs at", 2*pi, "radians.")
The zero occurs at 6.283185 radians.>
> cat("The zero occurs at", 2*pi, "radians.", "\n")
The zero occurs at 6.283185 radians.
> |
```

So, if I try to hear this here print you can see here it comes in the three different line, but in case if you want to write down here in a single line using the command cat. So, you can see here that it is coming in the same line. And, yeah, now if I try to show you what will happen if you do not try to use here this backslash n.

Now, you will see that as soon as you have executed this outcome came here and then this prompt sign that came on the next line, but now if you do not do it you can see here the prompt sign comes here, right. But, on the same line if you try to give it here the earlier command, then the prompt comes here on the next line. So, this is the use of this backslash n. So, this gives you the new line ok.

(Refer Slide Time: 10:42)



Formatting and Display of Strings

```
> d = date()
> cat("Today's date is:", d, "\n" )
Today's date is: Wed Dec 01 22:52:48 2021
```

```
> d = date()
> cat("Today's date is:", d, "\n" )
Today's date is: Wed Dec 01 22:52:48 2021
> |
```

[Please note this is the time at the time of preparation of slide]

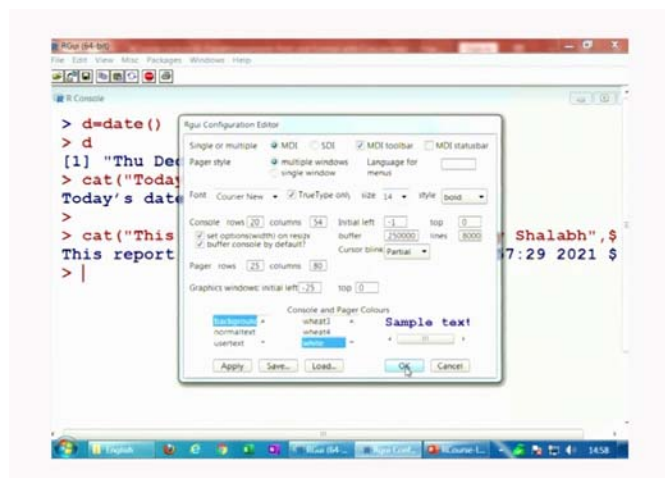
So, now, I come to our next example and try to show you. So, for example, you have learnt that there is a command in the R software to find out the time and date. And, suppose you want to write that today's date is and then after that whatever is the date or time that should come here. Actually these types of things will be very useful when you are trying to write down the report.

So, if you try to print this date and time with some statement, then people will very clearly understand that for example, if you write the report was printed on and then you try to give the command for the date or the function for the date. Then whenever the report is generated whatever is the date and time in the computer system that will be printed automatically there. For example, if you try to see here I try to use here variable here d to indicate the date. So, that was the command to find out the date and time.

So, I want to write here like this I want to write as a string that today's date is like this. So, I try to write down inside the double quotes and then I want to write down here whatever is the date today that should come. And, you know that both of them are going to be separated by the comma, but comma is not printed in the outcome and after that as a rule I will write down here backslash n.

Now, if you try to execute it, now you can see how the outcome will look like. Today's date is that is coming from here, then it is here the value of d Wednesday December 01 time is 22 hours 52 minutes 48 second 2021. Well, that is the time and date when I had prepared the slide, right and you can see here this outcome on the R software also.

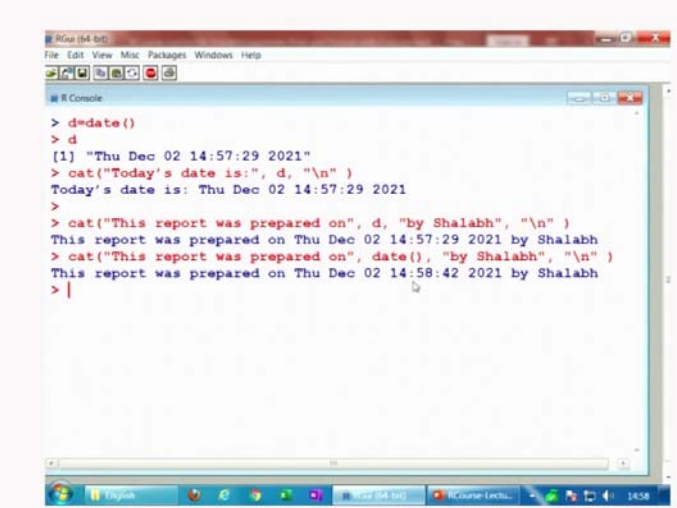
(Refer Slide Time: 12:27)



Now, at this moment if I try to do it here, then it is going to give you the time and date when I am recording this lecture, right. And, this is going to be different when you are trying to [Laughter] do it on your computer. For example, if you try to see here suppose if I try to see here d is your here date, right. So, you can see here the value of here d here is like this. Today is Thursday December 02 14 hours 57 minutes 29 seconds 2021 and if you try to have here a statement like this one today's date is like this and then d and then here backslash n.

Similarly, if you want to suppose I want to write here that this report was prepared on and after that I give you here date and time or after this also I can say here say by Shalabh, right and comma you can see here. This will come here like this, right if I try to reduce the phone size you can see it very clearly.

(Refer Slide Time: 13:06)



```
RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
> d=date()
> d
[1] "Thu Dec 02 14:57:29 2021"
> cat("Today's date is:", d, "\n" )
Today's date is: Thu Dec 02 14:57:29 2021
>
> cat("This report was prepared on", d, "by Shalabh", "\n" )
This report was prepared on Thu Dec 02 14:57:29 2021 by Shalabh
> cat("This report was prepared on", date(), "by Shalabh", "\n" )
This report was prepared on Thu Dec 02 14:58:42 2021 by Shalabh
> |
```

This is the outcome here this report was prepared on this today and yeah date is here like this yeah like this, right. And, if you want to make it here more general because now you can see here because d is coming directly, so, if you try to simply use here the command here directly this will also work.

So, you can see here the difference. If you are trying to give here the date directly here then when I executed the earlier command this was at 14:57:29, but now this command is at 14:58:42. So, this will give you the exact date and time when this report was

prepared, right. So, let us try to come back to our slides and try to see what else we have to learn, right.

(Refer Slide Time: 13:40)

```
Formatting and Display of Strings
> x = 1:10
> x
[1] 1 2 3 4 5 6 7 8 9 10

> cat(x, sep = " ++ ")
1 ++ 2 ++ 3 ++ 4 ++ 5 ++ 6 ++ 7 ++ 8 ++ 9 ++ 10 >

> cat("\n") # This is used to change the line

> cat(x, sep = " / ")
1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9 / 10
```

After this I try to give you here some combination of some numerical values and their separator. As the meaning of this separator suggest is that it is trying to separate two numbers or two strings etcetera. So, in order to explain it let me try to take here one example. I try to take here the number x 1 to 10, 1 2 3 4 5 6 7 8 9 10 and now I want to print this number, but you can see here between two numbers it is only here a blank. So, I do not want here blank, but I want to print here some number some symbol or whatever I want.

So, in order to do it the option here is that try to use the cat command. Try to give here the value or the data vector in which you have stored the values and then use the command sep; sep means separator then equal to. And, whatever you want to give in place of these blank spaces you try to give it within the double quotes.

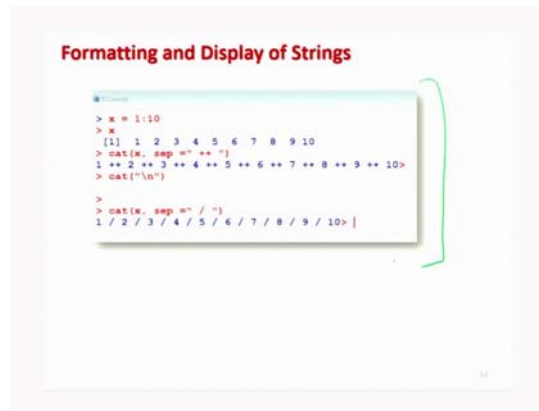
So, suppose you can see here I have given here blank space then two plus signs and then a blank space and I want that between any two number it should be like here this plus plus and then it should be here some here some here blank space.

So, now, if you try to do it this outcome will look like this 1 2 3 4 and you can see here between every pair of number this separator has inserted is inserted, right. And, yeah

because I am doing it continuously on the R console so, I am using here the command cat and within parenthesis within double quote backslash n. So, this is used to change the line.

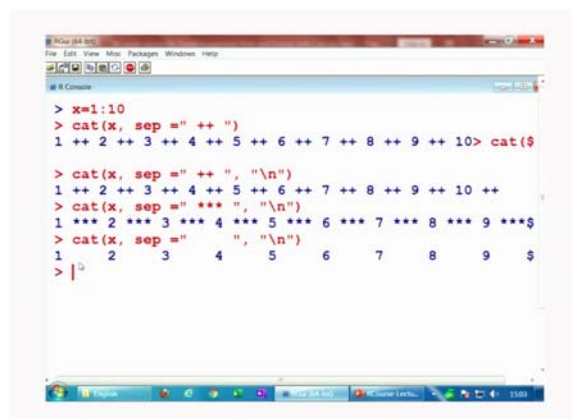
Otherwise if you do not do it your this prompt sign will come over here as I explained you in the earlier example. And, suppose I want to change this separator and I want to have here separator like as here this slash sign. So, I try to give it here blank sign then slash and then blank space. So, you can see here between any two numbers, this symbol whatever you have given inside the double quotes is entered here, right.

(Refer Slide Time: 15:36)



So, this is how the things work and this is here the screenshot of the same operation which I shown you. So, let me try to first show you these operations on the R console and then I move forward, ok.

(Refer Slide Time: 15:48)

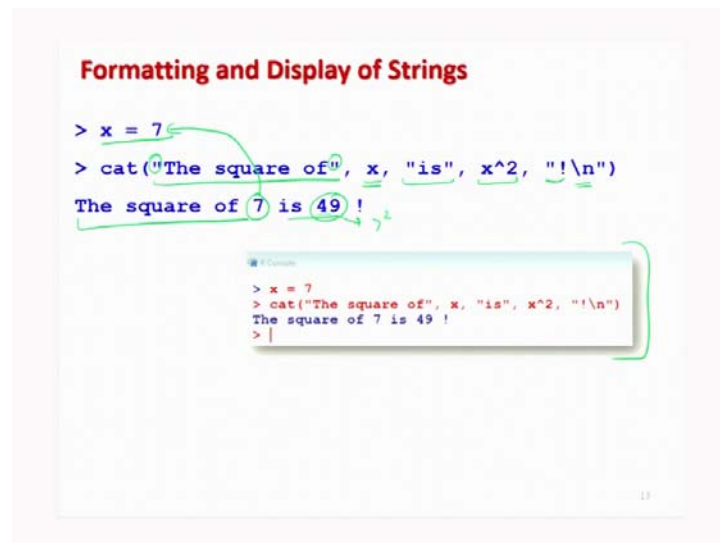


So, now if you try to see here I try to take here x equal to 1 to 10, and then I try to give it here like this. Now, you can see here 1 to 10, but this blank sign and plus, this is entered between every pair of the number you can see here. And, now you can see here your after this is executed the control is coming here, right. So, you have one option is that you try to take it here like to give here backslash n within the double quotes you using this cat command. So, it will come to the next line.

Or other option is this I can tell you which is a better option that you try to give here itself cat a cat is already there. So, you have to just give here backslash n inside the double quotes and now, you will see here this control will come to the next line you can see here instead of here, right.

Now, if I try to use here this command and suppose instead of here plus plus I try to give it here see three star. You can see here, now it is here like this and even if you want to give here only the blank spaces here like this you can see here now they are separated. So, this sep command is used to insert different types of separation with symbols or blank space within this command between two number, right.

(Refer Slide Time: 16:53)



```
Formatting and Display of Strings

> x = 7
> cat("The square of", x, "is", x^2, "!\n")
The square of 7 is 49 !

> x = 7
> cat("The square of", x, "is", x^2, "!\n")
The square of 7 is 49 !
> |
```

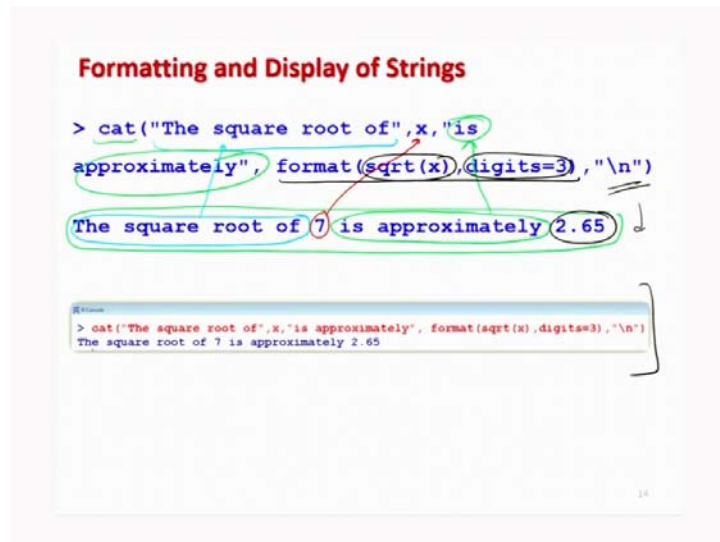
Similarly, I try to take here one more example. You have seen this type of examples in your mathematics books and I want to do it that I have here number here x equal to 7; this number is going to be changing, varying and it is user dependent. What I want that I

want to write that the square of this number is going to be whatever is the output of the square. So, I try to write down here the square of and this is a string.

So, I try to write down it within the double quotes and then I want to write down here the square of this number. Which number? I try to write down here x and then I try to write down here s within the double quotes and then whatever is the square of x which is x square I try to write down here it mathematically say x hat 2. And, after this I want to write down this exclamation sign and then next line.

So, if you try to execute it, now you can see what is happening. The square of 7; 7 is coming from where? Here is 49. 49 is coming from here 7 square and this is here the screenshot of the same operation, right. So, you can see here this is how you can control the output in the way you want, right.

(Refer Slide Time: 17:57)



Similarly, if I want to use here the format command also and suppose I want to have an outcome like this one. First you try to see here the outcome. The square root of 7 is approximately 2.65. So, if you want to do it now you have to look how you can frame your cat command. So, I try to write down here cat then the square root of this is a character. So, I try to give it here inside the double quotes.

And, after this, this is here a number 7 which is coming from here x. So, x is coming here and then after this you have once again pair character is and then it is actually is

approximately. So, this is approximately is given here within the double quotes and then after that you want here this 2.65 and your objective is that yeah if you try to see here square root of 7, that is going to take more number of digits. So, you want to restrict the output to have only 3 digit.

So, I try to write down here format and then inside the parenthesis sqrt square root of x and then digits is equal to 3. So, this is going to give you here this type of outcome and then yeah obviously, this backlash n will change the next line. So, and if you try to see here this outcome it will look like this. So, you can see here that whatever commands we have used here format and digits square root etc. that we already have learnt, right.

(Refer Slide Time: 19:17)

```
Formatting and Display of Strings

The cat function can also print simple vectors

> evenno = c(2,4,6,8,10)
> evenno
[1] 2 4 6 8 10

> cat("The first few even numbers are:",
evenno, "...\\n")
The first few even numbers are: 2 4 6 8 10 ...

R Console
> evenno = c(2,4,6,8,10)
> evenno
[1] 2 4 6 8 10
> cat("The first few even numbers are:", evenno, "...\\n")
The first few even numbers are: 2 4 6 8 10 ...
>
```

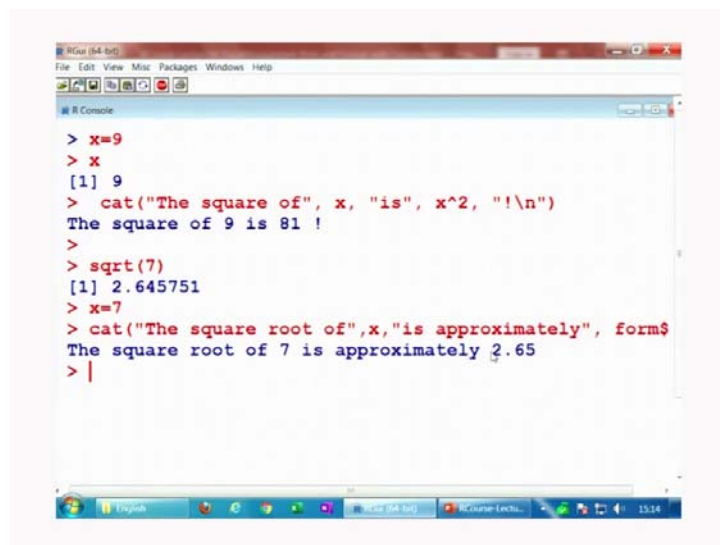
Now, in both these examples I have taken the input variable x as a single value a scalar. Now, I try to take here a data vector for example, if I try to take here a data vector of 5 values 2 4 6 8 10 and let us call it as a even number that is even and odd. So, this is this data vector. Now, I want to print like this. So, instead of looking at the command first, you try to look at the outcome that this is what we want and now you have to think how you have to write the command.

So, if you try to see here the first few even numbers are this is a character. So, it is written here inside the double quotes and then you want to print like this 2 4 6 8 10 etcetera. So, this is coming from where? From this even number. So, you try to write

down only here like this e v e n n o and then comma. And, then after this if you try to observe you want to write down here dot dot dot this means continued.

So, for that this is also character. So, you try to write down here like this here dot dot dot and then the action to change the line and you will get here this outcome that you can see in the screenshot here. So, you have seen this type of outcome many times they are printed in the mathematics books and in different types of reports. So, now, you can see it is not a very difficult thing to do, right.

(Refer Slide Time: 20:42)

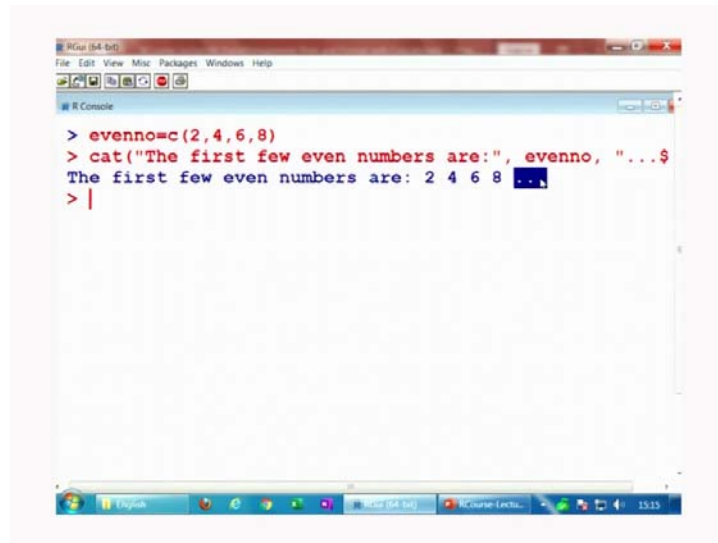


```
RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
> x=9
> x
[1] 9
> cat("The square of", x, "is", x^2, "!\n")
The square of 9 is 81 !
>
> sqrt(7)
[1] 2.645751
> x=7
> cat("The square root of",x,"is approximately", format$
The square root of 7 is approximately 2.65
> |
```

And, before I move forward let me try to give you here these things how do they work in the R console. So, I try to take here some value here x equal to suppose here 9. So, this is here x is 9 and if you try to print here this command the square of x is x square, so, what you get here? The square of 9 is 81. Similarly, if you try to see here what is the value of a square root of say 7 you can see here this is 2.645751.

And, if you try to take it here now here x equal to here 7 and if you try to use here this command here that the square root of x is approximately format square root of 27 square root of 7 it will come out to be the square root of 7 is approximately 2.65. So, this is an outcome of the statement, right.

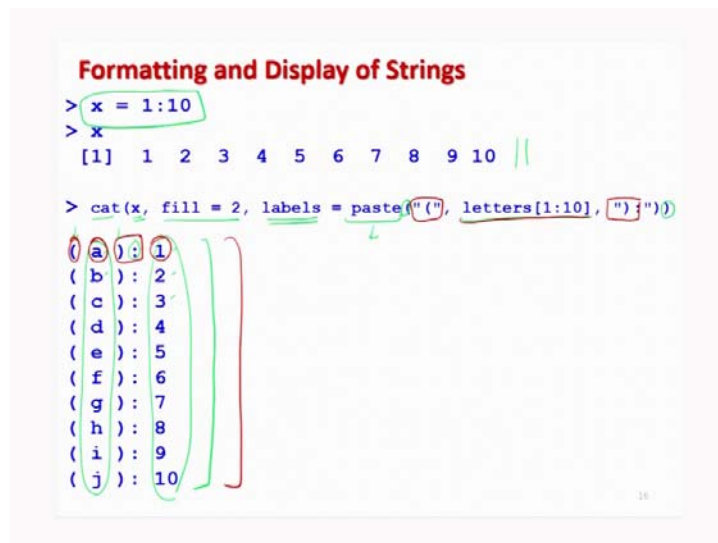
(Refer Slide Time: 21:26)



```
> evenno=c(2,4,6,8)
> cat("The first few even numbers are:", evenno, "...")
The first few even numbers are: 2 4 6 8 ...
> |
```

And, similarly if you try to take here this example where you are trying to take some here this data vector so, if I try to say here even number is equal to c 2, 4, 6, 8 like this. So, if you try to press here like this then it will give you the first few even numbers are 2, 4, 6, 8 after dot dot dot, right, ok.

(Refer Slide Time: 21:41)



```
Formatting and Display of Strings
> x = 1:10
> x
[1] 1 2 3 4 5 6 7 8 9 10 ||

> cat(x, fill = 2, labels = paste("(", letters[1:10], ")");))
(a): 1
(b): 2
(c): 3
(d): 4
(e): 5
(f): 6
(g): 7
(h): 8
(i): 9
(j): 10
```

Now, I give you one more example that suppose I want to do a very specific type of operation like this one. First you can see here at the outcome. So, what is happening here? You have here this number 1 2 3 4 up to here 10, then you have here alphabets in

lower case a b c d up to here j. And, you want to print them like this that there is here a parenthesis inside the parenthesis this is here are the alphabets and after this there is a colon and then this here one.

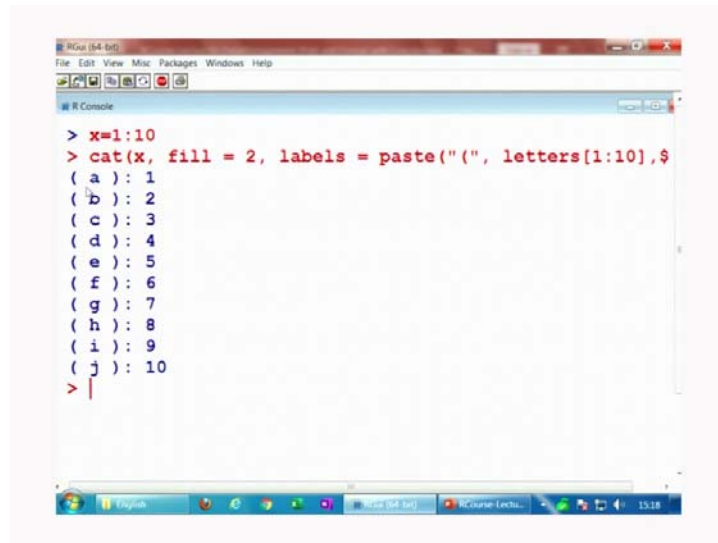
So, how to write down here? How to write this thing using the fill command here and with the cat. So, first in order to generate here this numbers 1 to 10 like here this one I try to use here the command `x is equal to 1 to 10; 1 colon 10`. So, this will give me here this 10 values now I want to generate first this here lowercase alphabet. So, I try to give here the I try to use here the command `letters` in lower case `letters` and within the square bracket 1 to 10, right.

After this you have to how you are going to print here. So, if you try to use your cat then what you want to print `x` and then this fill is equal to 2 and then now, you have to give here the labels and labels you want to hear paste. Well, we have not done the command paste here, but in the forthcoming lecture we are going to learn about it and after that you can revise this example once again and then you will understand it very clearly, right.

So, if you try to see here what I am trying to do here, this is a parenthesis for the command paste. So, no issues. So, now, whatever I have to write I am writing the in this color red. So, whatever is this here this parenthesis this is now given here like this inside the double quotes and after that you want here in letters for example, this is here a. So, you are going to write down here `letters 1 colon 10` and after that what you want here is this number here colon and can hear this number 1.

So, this colon is and actually you want here this bracket and colon together. So, this bracket and colon together they are here like this, right and then you can see here you want here `x`. So, this is coming here number. So, now, if you try to see this outcome will look like this. So, let me try to show you it on the R console and then you will understand it very easily, right.

(Refer Slide Time: 23:55)

A screenshot of an R console window. The window title is "R Console". The command prompt shows the following code:

```
> x=1:10  
> cat(x, fill = 2, labels = paste("(", letters[1:10], $
```

The output of the command is displayed as a list of pairs:

```
( a ): 1  
( b ): 2  
( c ): 3  
( d ): 4  
( e ): 5  
( f ): 6  
( g ): 7  
( h ): 8  
( i ): 9  
( j ): 10  
> |
```

The console window has a menu bar with "File", "Edit", "View", "Misc", "Packages", "Windows", and "Help". The taskbar at the bottom shows the Windows logo, a search bar, and several open applications including "RStudio (64-bit)", "RStudio - Lect...", and a clock showing "15:18".

So, if I try to take here this command here x like this you can see here x is equal to here 1 to 10 and then here like this you can see here you are getting here this type of value. But, yeah I agree at this moment because I have not told you about the face command. So, it may not be 100 percent clear, but anyway in the next lecture I am going to talk about it. So, after this it would not be difficult for you to understand this command over here and this example here, right.

So, now we come to an end to this lecture and you can see here that in this lecture that was very simple and I have explained you the use of the cat function and with this cat function you have an advantage that you can join the things together and which was not possible in the print command and in cat there is a possibility that you can combine different types of data object like as you have combined the character and number character means some statement and the number means that can be input that can be output or that can be some function also the value of some function.

So, now this opens a lots of opportunities for you to create example and try to practice. You can just look into any book or that or any report from the software how they are trying to generate it and then you try to see how you can generate the same report in R software. Well, there are some paid software which generates this type of report automatically you just say click and after doing the analysis whatever you want and then they will generate a report which is very user friendly.

The only difference in the R software is that you need to understand what type of report you want where you want to write what and then you have to use this print, cat, paste and different types of formatting commands to generate the exactly the same report and that is without any cost. But, you have to work, you have to write the program. So, you try to practice it and I will see you in the next lecture.

Till then, goodbye.