

## **Lecture 02**

### **Calculations with R Software – Basics and R as a calculator**

Welcome to the, next lecture on, Descriptive Statistics, with R software. You may recall that, in the earlier lecture we had a quick review about the R Software and the related information. Now, in this lecture and in the next couple of lectures, I'm going to give you, an idea about the basic operations or the basic mathematical operations, in R Software. And these are the operations which we are going to use again and again in the forthcoming lecture. So, this will be a quick introduction to them minimum basic required this mathematical operators and how to use them, in our, in this lecture and in the next couple of lecture.

Refer Slide Time :( 00:59)

**Basics**

- **>** is the prompt sign in R.
- The assignment operators are the left arrow with dash **<-** and equal sign **=**.

**> x <- 40** assigns the value 40 to **x**.

**x <- 40**

**x = 40**

**> x = 40** assigns the value 40 to **x**.

Initially only **<-** was available in R.

```
R Console
> x <- 40
> x
[1] 40
```

```
R Console
> x = 40
> x
[1] 40
```

So you can see here, as soon as you start the R Software, on the console you get a sign like greater that (>) sign. This I explained you in the earlier lecture. This is the prompt sign. This is the prompt sign on the console, after which you try to write down the commands, for their execution. You type the command, after the prompt sign and press enter, the command will be executed. So one I would like to clear you here, that when we are trying to assign a value to a variable, in mathematics we always write. Suppose I want to assign a value 2, to a variable say, x, then I would write here, x = 2. The same thing is followed in R also. Suppose I have a variable x and I want to assign here the value to x, then I would write here, say x = 2. But before that, I would try to explain you one thing more. When this R has started, at that time, the initial assignment operator was not this equality sign. But this was less (<) than and hyphen (-) sign. So in the older versions of R, once we try to assign a value, I have to write, less than and hyphen.

Refer Slide Time :( 03:08)

## Basics

- `>` is the prompt sign in R.
- The assignment operators are the left arrow with dash `<-` and equal sign `=`.

`> x <- 40` assigns the value 40 to `x`.

`x <- 40`

`x = 40`

`> x = 40` assigns the value 40 to `x`.

Initially only `<-` was available in R.

```
> x <- 40
> x
[1] 40
```

```
> x = 40
> x
[1] 40
```

For example, if I want to assign a value 40 to a variable `x`, then I need to write down here `x <-` and here 40, like this. But now in the recent versions of R Software, they have used the equality sign, so either I try to write down here, `x = 40` or say, `x <- 40`, both are acceptable, in the recent versions of R. So that you have to, just keep in mind. So I would try to show you here, how this things are happening and what does this mean, when I say that I'm trying to assign a value `x=40`.

Refer Slide Time :( 03:10)

```
File Edit View Misc Packages Windows Help
R Console
> x <- 40
> x
[1] 40
>
> x=40
> x
[1] 40
> x=20
> x
[1] 20
> |
```

So, suppose if I try to show you here, if I try to copy this command and I try to paste it on the R console, you can see here, this is giving me, now I press here `x` and in turn, this is giving me `x=40`. And similarly if I try to write down here, `x=40`, using the equality sign, then also `x` is giving me here

40. Or I can take here another example, just to discriminate between the two, if I write  $x=20$ , now if you see here,  $x$  becomes here, 20.

Refer Slide Time :( 03:36)

**Basics**

>  $x = 40$  assigns the value 40 to  $x$ .

>  $y = x * 3$  assigns the value  $3*x$  to  $y$ .

$y = x * 3$

>  $z = x - y$  assigns the value  $x - y$  to  $z$ .

$z = x - y$

R Console

```
> y = x * 3
> y
[1] 120
```

R Console

```
> z = x - y
> z
[1] -80
```

And then once you are trying to assign a value to a variable, then there are two options. You can assign, a numeric value to the variable, as well as, you can assign, a variable to variable also. For example, if I say here, suppose I assign here  $x=40$  and now I say, I will multiply this  $x$  by 3 and whatever is the outcome, this I will try to store in the, in say, another variable here,  $y$ . So I can write also here,  $y=x*3$ .  $X$  star ( $*$ ) means, star operator is the multiplicative sign. Where which I will explain after couple of slides also. And similarly the minus ( $-$ ) signs is the differences operator. So similarly if I want to multiply  $x$  by 3 and I want to store in a variable  $y$ . I can write down here  $y = x*3$ . And similarly, in case if I want to find out the difference between  $x$  and  $y$ , that means I want to find out  $x-y$  and suppose I want to store the values of  $x-y$  in say another variable, say  $z$ , then I would write down here,  $z=x-y$ , like as here. Right, I can show you here, on the R console, that how the things are happening.

Refer Slide Time :( 05:03)

```
File Edit View Misc Packages Windows Help
R Console
> x=20
> y=3*x
> x
[1] 20
> y
[1] 60
> z=x-y
> z
[1] -40
> |
```

So I try to show you here, suppose if I take here,  $x=20$  and now I say  $y=3$  into multiplied by  $x$ . So, if I show here, the value of  $x$  here is, 20 and the value of here,  $y$  comes out to be 20 into 3, this is 60. And suppose I try to find out the value of  $x-y$  and I try to store it in, say, another variable here,  $Z$ . So if I try to write down here,  $z=x-y$  and if I try to, see the value of  $z$ , this comes out to be minus 40, that is  $20 - 60$ . So you can see here, this is how; we are going to assign or store a value inside a variable.

Refer Slide Time :( 05:48)

## Basics

- **#** The character # marks the beginning of a comment.

All characters until the end of the line are ignored.

```
> # mu is the mean
> # x = 40 is treated as comment only
```

Now if you try to see here, there is another symbol here, Hash (#). **This hash is used to** indicate, that the given command, on the command line, is only a comment. It has not to be executed. And once you write there, Hash (#) and followed by whatever you want to write, any command or any syntax,

nothing is going to be executed and this will be taken as the comment and all mathematical operator after that will be ignored.

Refer Slide Time :( 06:20)

```
> x=20
> y=40
> z=x-y
> z
[1] -20
> #m=x-y
> m
Error: object 'm' not found
> |
```

Let us consider, say, x is equal to say, 20, and say, y equal to here, 40, once again I try to define here,  $z=x-y$ , which is here, giving me the value -20, that it 20-40. Now I define here, another value here, say here, m and I try to put here Hash (#) sign. And I try to say here, Hash (#) $m=x-y$  and you will see here, there is no value here, something called here, 'Error'. Because, it has not been executed. This is only stored inside the R console, as a comment. And the comment cannot be used. Right. The use of this comment sign is this. When we are trying to write down a longer program, we try to explain the features and the names of the variable, inside the program, so that after sometime, when we forgotten about it, by looking at this comments, I can recall, what was the meaning of x, what was the meaning of y or say, what was the meaning of z.

Refer Slide Time :( 07:28)

## Basics

- Capital and small letters are different.

> X = 40 and > x = 40 are different

*x = 40*  
*X = 40*  
*↳ different*

```
R Console
> X = 40
> X
[1] 40
```

```
R Console
> x=40
> x
[1] 40
>
> X
Error: object 'X' not found
>
> X=30
> X
[1] 30
> x
[1] 40
```

Now, the next aspect. A very important point in using R is this. There is a difference in the use of small letters, alphabets and capital letter, alphabets.

For example, incase if I try to use here, small x, say equal to 40 and if I try to write down here X capital X = 40, then these two are different. Right? I can show you here, with an example, on the R console itself.

Refer Slide Time :( 08:01)

```
R Console
File Edit View Misc Packages Windows Help
> x=40
> x
[1] 40
> X=70
> X
[1] 70
> x
[1] 40
> X-x
[1] 30
> |
```

For example, if I'm trying to, use here, some command here, say x equal to here 40. Now you can see here, the value of x stored here, is 40. But, now in case if I say here, capital X, capital X is equal here is equal to 70, then once I say here, capital X, this will give me here the, 70 and if I try to recall the value of small x, this is 40. And even I can find out the value of here, x minus small x, this capital X minus small x, this will come out to be, 70 minus 40, this is 30. So you can see here, that this small and capital letter alphabets, have different types of interpretation and here,

Refer Slide Time :( 08:44)

**Basics**

- Capital and small letters are different.

> X = 40 and > x = 40 are different

*x = 40*  
*X = 40*  
*different*

```
R Console
> X = 40
> X
[1] 40

R Console
> x=40
> x
[1] 40
>
> X
Error: object 'X' not found
>
> X=30
> X
[1] 30
> x
[1] 40
```

this is the, screenshot of the same thing, which I have done here. But, yeah I mean, I have used a different values, but that is the same thing here. Right. Okay. Now another the thing statistics is that, you will always be dealing with the data. That is why you are doing a computation. So in a statistics and in using R, whenever we want to input the data, I always have to write here, the letter, c.

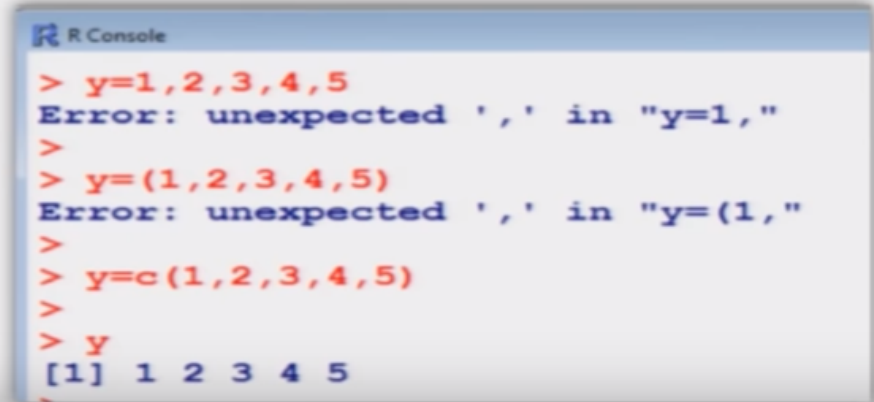
Refer Slide Time :( 09:12)



## Basics

- The command `c(1,2,3,4,5)` combines the numbers 1,2,3,4 and 5 to a vector.

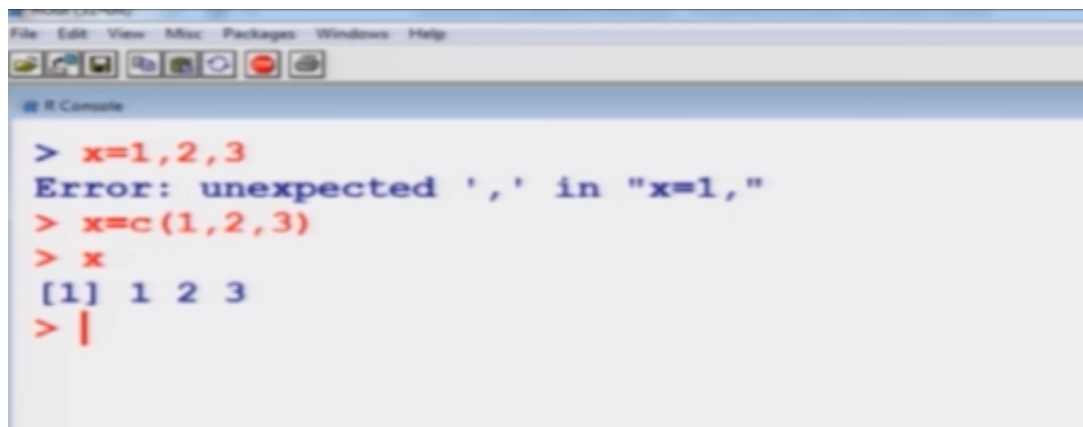
`c(1,2,3,4,5)`



```
R Console
> y=1,2,3,4,5
Error: unexpected ',' in "y=1,"
>
> y=(1,2,3,4,5)
Error: unexpected ',' in "y=(1,"
>
> y=c(1,2,3,4,5)
>
> y
[1] 1 2 3 4 5
```

Suppose I want to write down the values, 1,2,3,4,5, then I need to input the data in R, by writing here, c, small c, not capital C. And the 1 comma, 2 comma, 3 comma, 4 comma, 5, that will, all the values are separated, by a comma (,) operator. And in case if you don't use the c command here, then it will give you, different types trouble. For example, I can show you here it, on the R console here.

Refer Slide Time :( 09:43)



```
R Console
File Edit View Misc Packages Windows Help
> x=1,2,3
Error: unexpected ',' in "x=1,"
> x=c(1,2,3)
> x
[1] 1 2 3
> |
```

Now, let me choose here, x=1, 2, 3. Now you will see here, it is giving me an error. But incase if I try to use here, x=c(1, 2, 3), then it is giving me here, x as here, 1 2 3 . Right. Okay. So, always remember to give the values of the data, you closing the, c operator.

Refer Slide Time :( 10:17)

## R as a calculator

**Power**  $3^2$

```
> 3^2  
[1] 9
```

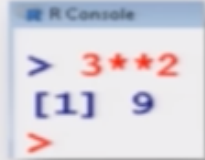
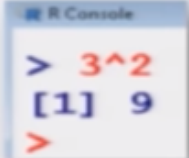
# Command  
# Output

```
> 3**2  
[1] 9
```

# Command  
# Output

$3^2$   $3^2$

$3^2$   $3^2$   
 $3 \times 2$



The image illustrates two ways to calculate 3 squared in R. The first method uses the caret operator (^), and the second uses the double star operator (\*\*). Both commands result in the output [1] 9. Handwritten green annotations show the mathematical notation 3^2 and a diagram connecting the two R commands to this notation. A box containing 3^2 is also shown with arrows pointing to the handwritten 3^2 and 3 x 2.

Now I will come to the basic operation, that incase if you want to do the addition, then the operator here is, plus (+) sign. The usual mathematical sign, you have to use the, plus operator here. For example, first I will try to show you, the outcome and then I will show, you on the R console also. Suppose if I want to add, 5 and 3, then I simply have to give here, 5 and 3, and then here, the operator plus (+). So I simply have to type here, 5+3 and this will give me here, the value, 8. So, the, the, the, this is the screenshot. And similarly, if you want to multiply 5 and 3, then you have to write the, 5 multiplied by 3. So the multiplication operator here is, the star (\*). So I would simply write it here, 5\*3 and this will give me here a value here, 15. And this is here, the screenshot. So, similarly if you want to subtract two values, the mathematical operator is the same, that is, so this subtraction sign (-), hyphen. And if I want to subtract, 5 and 3, then I have to write here, 5-3, in the same way, as we do in the usual mathematics and this will give me the value here, 2 and this is here the screenshot. And, incase if I want to use the division, then the division operator is backslash (/) and incase if I want to divide, 5 by 2, then I simply have to write down here, 5/2 and you can see here, the answer will come out to be, 2.5 and this is here, the screenshot, of the operator.

And similarly incase if I want to write down here, 3 square. Right. So, incase if I want to write down here 3 square, then I have here now, two options. I can write down here, say 3, say here at (^) 2 or I can write down here 3 double star (\*\*). So, you can see here, I have used here two commands. This and this here, two indicate the same value, 3 square. So whenever you want to write any mathematical equation, related to the power, then I have two option. One is to use the at (^) sign

another the thing is to use the double star (\*\*) sign. And 3 square in both the case will give 9 and these are the two operations here, which I will show you very soon.

Refer Slide Time :( 12:45)

### R as a calculator

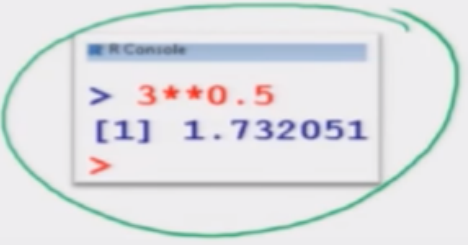
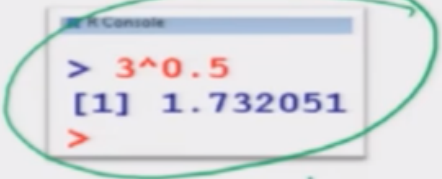
**Power**

$\sqrt{3} = 3^{1/2} = 3^{0.5}$   $\left\{ \begin{array}{l} 3^{0.5} \\ 3 \times \times 0.5 \end{array} \right.$

```
> 3^0.5 # Command
[1] 1.732051 # Output
```

```
> 3**0.5 # Command
[1] 1.732051 # Output
```

$3^{1/2}$



The image shows a slide titled "R as a calculator" illustrating two ways to calculate the square root of 3 in R. Handwritten green notes show the mathematical equivalence  $\sqrt{3} = 3^{1/2} = 3^{0.5}$  and a bracketed list of  $3^{0.5}$  and  $3 \times \times 0.5$ . Two R console screenshots are shown, both displaying the command and output for  $3^{0.5}$  and  $3^{**0.5}$ , both resulting in the value 1.732051. A small box at the bottom left contains the mathematical expression  $3^{1/2}$ .

And similarly, whence, once you are trying to giving the power, this power can be an integer or it can be a fraction value. Also for example, in case if I want to find out the value of the square root of 3, the square of 3, is only 3, is to the power of, 1 by 2, which is equal to a 3, 0.5, 3 is to power of 0.5. So I can write down this value here as, say, 3 at(^) 0.5 or 3 \*\*0.5, which I have done here. And you can see here in both the cases, the values are coming out to be 1.73 and this is, and these two are, the screenshot which have been obtained after this education.

Refer Slide Time :( 13:32)

## R as a calculator

### Power

```
> 3^-0.5 # Command  
[1] 0.5773503 # Output
```

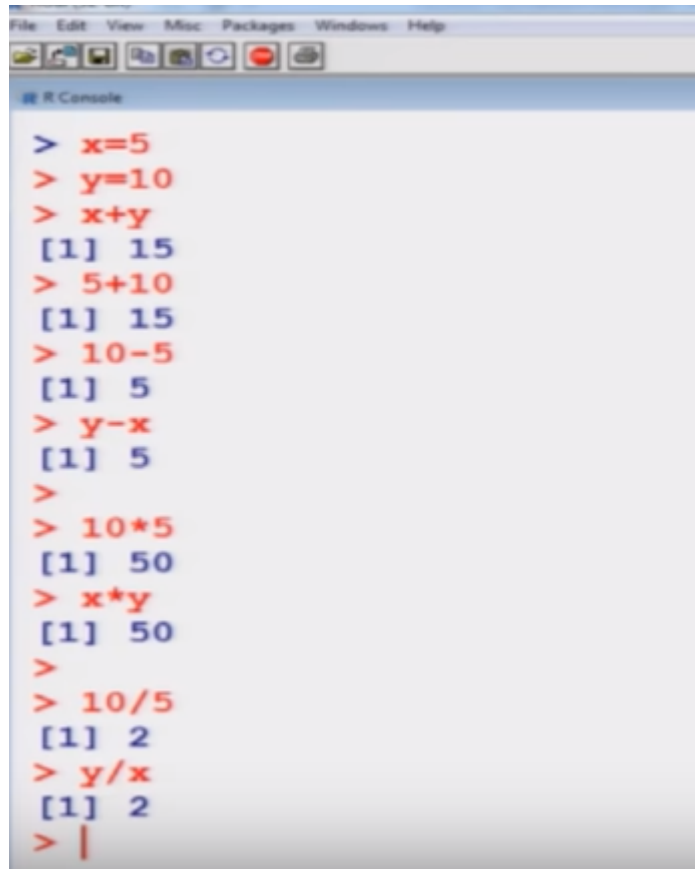
$$\frac{1}{\sqrt{3}} = 3^{-1/2} = 3^{-0.5}$$

$$3^{-1/2}$$

```
R Console  
> 3^-0.5  
[1] 0.5773503  
>
```

And similarly, in case if you want to find out the value of 1 upon square root of 3, so this is going to be  $3^{-1/2}$ , this is 3 to the power of minus 1 upon 2, this is 3 to the power of minus 0.5 and it can be written, either by putting the at (^) sign or by using the double star (\*\*) sign. For example here, I have written this thing and here you can see here, this is the screenshot here. And, yeah, in case if you have more than one operator at the same time. Suppose you are trying to use the plus (+) sign, minus sign (-), multiplication sign (\*), division sign (/), then using the same rule, what we have the Bodmas rule, the same Bodmas rule is applicable, in case of, R Software also.

Refer Slide Time :( 14:26)

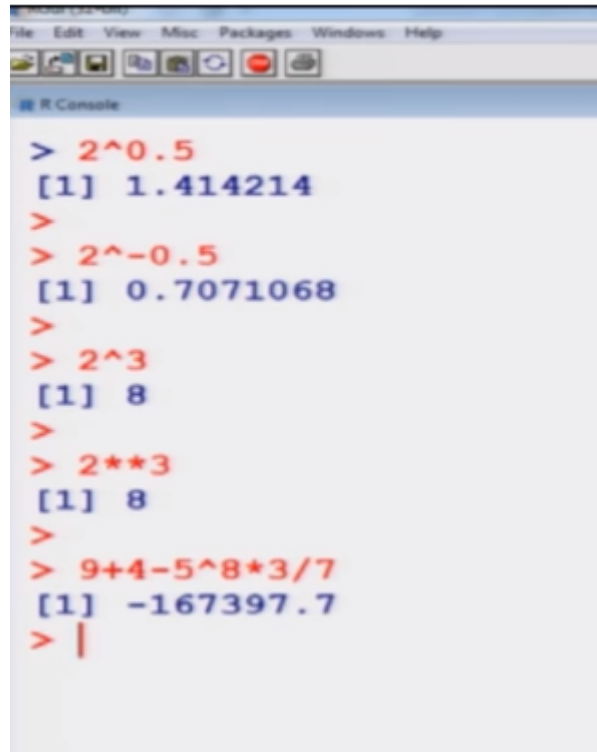
A screenshot of an R console window. The window has a menu bar with 'File', 'Edit', 'View', 'Misc', 'Packages', 'Windows', and 'Help'. Below the menu bar is a toolbar with several icons. The main area of the window is titled 'R Console' and contains the following text:

```
> x=5
> y=10
> x+y
[1] 15
> 5+10
[1] 15
> 10-5
[1] 5
> y-x
[1] 5
>
> 10*5
[1] 50
> x*y
[1] 50
>
> 10/5
[1] 2
> y/x
[1] 2
> |
```

So for example, now I would try to show you all these things over the R console here, so that you have some confidence here. For example, I can use here, x equal to here 5, y is equal to here, say 10, so now, in case if I want to add it, I can write down here,  $x+y=15$  or in case if you want to write down here directly, say  $5+10$ , it will also give you the value here, 15.

So similarly, in case if I want to, subtract it, say  $10-5$ , so  $10-5$  will be a, see here, if you can see here, 5 and suppose if, I give here the value here, also y minus here x, so this will also be here, 5. And similarly, in case if I want to multiply 10 and 5, so I have to write down here  $10*5$  and this will give me the value here, 50. And similarly if I want to multiply, x and y, which are taking the same values, this will again give me the value, 50. Now, in case if I want to divide, 10 by 5, so I have to write down,  $10/5$  and will give me the value then, 2 and similarly if I want to write down here y divided by here x, this will again give me the value here, 2.

Refer Slide Time :( 15:48)



```
> 2^0.5
[1] 1.414214
>
> 2^-0.5
[1] 0.7071068
>
> 2^3
[1] 8
>
> 2**3
[1] 8
>
> 9+4-5^8*3/7
[1] -167397.7
> |
```

And similarly, I clear this screen, by pressing control L (Ctrl L). And suppose I want to find out the, square root of 2, then I have to give it here, say a 2, say at (^), 0.5, so this will give me here, 1.414214. And similarly if I want to find out the value of 1 upon, square root of 2. Then I have to give here the value here,  $2^{-0.5}$  and this will give me the value of here, 1 upon, square root of 2. And similarly, if you want to find out here the value of 2 cube, then I have two options here. I can write down here,  $2^3$ , this will give me here the value here 8 and similarly, if I write down here,  $2^{**}3$ , this will also give me the value here, 8. Right. And similarly incase if I have, any other mem, mathematical operator like as,  $9+4-5^8*3/7$ , so this is again the value, which using on the rule of Bodmas. These are very simple operators, but definitely, before we go into the statistical part, you need to practice them, so that you are more conversant. So, I would request you, do please practice them. And I will see you in the next lecture. Till then. Goodbye.