

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

**Basics of Calculations**  
**Lecture - 24**  
**Functions**

Hello friend, welcome to the course Foundations of R software. In this lecture we are going to learn about a new topic this is about Functions. So, the first question comes here what is a function? Instead of asking you what is the function if I ask you that do you know what is the program is in terms of the computer language, I am sure you all know what is a program.

So, now if I say that whatever is the meaning of the program in any other computer language that is called as a function in the R software. After this do you think that it is very difficult is it very difficult to understand what are we going to do today not at all. I am sure that you must have done some programming in some other languages and so, whatever you have done earlier, now we want to know that how it can be done in the R software.

And the programming in the R software is one of the strongest tool of this R software that you can write a program and within the program also you can call another program and that is why this R programming is very strong.

So, once you try to define a function then within the function also you can define functions and so, very complicated jobs can be programmed very easily in the R software. So, first we try to understand what is this function and I will try to give you a very interesting illustration that how to write this program just in a couple of minutes and I can promise you know it ok.

So, let us begin our lecture and try to understand what is the function, but before that do you remember what was the function when you studied the function in class 10, class 11 or class 12? Try to recall the function  $y$  equal to  $f x$ .

(Refer Slide Time: 02:26)

**Functions**

$y = f(x)$   $x \rightarrow$  input variable  
 $y \rightarrow$  output  
 $y$  is a function of  $x$

$f = x^2$   
 $y = x^2$   
 $x = 100, y = 100^2$

$y = \text{function}(\text{input variables}) \{ f \}$   
 $z = x^2 + y^2$  Input  $x, y$   
Output  $z$   
 $x = 2, y = 3, z = 2^2 + 3^2$   
 $f \rightarrow$  Square & sum

$f$   
 $\downarrow$   
known

So, let us begin our lecture with this  $y$  equal to  $f x$ . So, do you remember that in your high school class 11 or class 12 you used to write down a statement like  $y$  equal to  $f x$ . And if you try to write down the statement then you will simply write this that  $y$  is a function of  $x$  exactly that is the same meaning which has been transformed into the R software and this is the same language we try to use here.

So, now first you try to understand what are the ingredients of a function. In this function you can see here there are two values  $x, y$  and  $x$  here is the input variable and whatever you input based on that you get an output here which is indicated by  $y$  and how this input and output are going to be controlled? This is by here  $f$  and this  $f$  is known to us that what exactly we want to do for example, if you try to write down here function here like as  $x$  square.

So, this this means here  $y$  is equal to here  $x$  square so; that means, whatever input value you are going to give here say  $x$  equal to suppose 100, then  $y$  is going to be 100 square right. So, in order to write a function in the R programming we follow the same setup. I will simply write here  $y$  is a function like this  $y$  equal to function and then inside the parenthesis, I will try to write down here the input variables there can be one variable there can be more than one variable.

So, I try to list all of them here and then after this whatever execution I want to do or whatever is the form of here f, this I will try to write down inside the curly bracket that is all right. Similarly, if you try to say if I try to write down here z is equal to x square plus y square the this means here what?

Your input variables are here x and y and your output variable here is z so; that means, if you try to take here x equal to 2, y equal to here 3, then z is going to be here 2 square plus 3 square and your f is like here a square function square and then sum like this. So, this is the same thing I am going to do here in this lecture today. So, let us begin our lecture and try to understand it.

(Refer Slide Time: 05:03)

**Functions**

- Functions are a bunch of commands grouped together in a sensible unit
- Functions take input arguments, do calculations (or make some graphics, call other functions) and produce some output and return a result in a variable. The returned variable can be a complex construct, like a list.

So, the first question comes here what is the function. So, functions are simply like program and what are programs? Programs are the bunch of commands which are grouped together in a sensible way so, that they are executable inside the software or the programming language and these programs are written to execute a particular task and they are written in a sequence which makes sense and they give you the same output what you really want and for writing the program you can use all your concept means mathematical operator, logical operator, conditional execution etc.

So, what does this functions do? They take input arguments, they will do the calculations or make some graphics call other function whatever you want and they will try to

produce some output and whatever is the output that is the result which is obtained after executing the function and the written value whatever is the result this can be a complex construct like a list, data frame etc. also what are listed data frame etcetera that we will try to discuss in the forthcoming lectures.

(Refer Slide Time: 06:18)

**Functions : Components**

**Function Name** - This is the name of the function which is stored as an object with this name.

**Arguments** ( ) - An argument contains the input values. A function may contain no arguments.

**Function Body** f - Contains statements that defines what the function has to do.

**Return Value** - The evaluated value of the last expression in the function body.

So, now we have understood that when we want to write a function or a program, but now I will use the word function instead of program. So, you need to have a couple of ingredients the first thing is this you need to define a name of the function. So, this is the name of the function which is stored with this name for example, you try to give a program a name right then arguments. So, this arguments; that means, within parenthesis you have to write all the input value values.

And it is possible that the function may contain some values inside the argument or parenthesis or it may not contain any value inside the parenthesis because sometime if you are trying to give the input value during the program or the input is coming from some other program, then possibly you may not need to write down these values.

But otherwise you have to write all the input variable inside the parenthesis, then the function body it is something like your here f. So, this body contains all the statement that define what the function has to do, what is to be done, right and after that the

outcome. The outcome is the return value that whatever calculation evaluation have been done up to the last expression of the function body they are written here as an outcome.

(Refer Slide Time: 07:46)

**Functions : Built-in functions**  
R has Built-in functions also.

Simple examples of in-built functions are  
sum(), prod(), mean(), max(), sum(x) etc.

5

So, now we have a two options we can write our own functions or we can use the built in functions also. So, built in function for example, you have read they were like sum, product, mean, maximum, sum etc. So, they say built in functions are also the functions which were written by someone else and they have been given the name of the function which we try to see as the name of the built in function.

Somebody had written the program for the sum and it has given the name as sum someone has written the function for finding out the product and it has been given the name prod and so on. So, these functions are built in we which are available inside the R software or you can create yourself also.

So, we are going to learn here today that how you can create a program although we are going to discuss here very simple example, but these examples can be extended to any level that is my promise to you if you understand it and when you are trying to write down the program you can also use this built in functions also without any problem and that is the beauty of this R software, right.

(Refer Slide Time: 08:58)

### Functions

Syntax

```
Name <- function(Argument1, Argument2, ...)
{
  expression(s)
}
```

where `expression(s)` is a single command or a group of commands.

You can use `=` operator also.

```
Name = function(Argument1, Argument2, ...)
{
  expression(s)
}
```

So, as I explain you the structure of writing the function is like this that first you try to give a name to the function. And then you try to write down the equality operator. So, you can use this operator or you can use this operator also you can see here I have given it in the bottom of the slide the same thing with the equality operator.

After that you have to write function `f u n c t i o n` in lower case alphabets and then within parenthesis within Arguments you have to give all the input variables. You have to write down here the list of all the input variables which are required to execute the syntax and command that you are going to write further. So, this Argument 1, Argument 2 they are the name of the input variable I am trying to indicate like this. After this you try to write down the curly bracket and within this curly bracket try to write down all the expression commands etcetera whatever you want to execute.

So, these expressions can have more than one command they can also have single command also. So, that depends on your need and the and when you are trying to give it here a name that instead of using the operator less than hyphen, you can also use here the equality side that will not make any difference, right, ok.

(Refer Slide Time: 10:20)

### Function arguments with description and default values

- Function arguments can be given a meaningful name
- Function arguments can be set to default values
- Functions can have the special argument '...'

7

So, now, before you try to go for creating a function, let me try to give you some tips. The always try to give function arguments a meaningful name and function argument can be set to some default values also and they may and they can also have some special arguments also.

(Refer Slide Time: 10:42)

### Functions (Single variable)

The sign = or <- is furthermore used for defining functions:

```
> abc = function(x) {  
+   x^2  
+ }  
# Console  
> abc = function(x) {  
+   x^2  
+ }  
> abc  
function(x) {  
  x^2  
}
```

A function is called with argument as

```
> abc(3) ← x=3  
[1] 9  
> abc(6) ← x=6  
[1] 36  
> abc(9) ← x=9  
[1] 81
```

$f(x, y)$   
 $x=2, y=3$   
 $f(2, 3)$   
If you write  $f(3, 2)$   
 $x=3, y=2$

8

So, let us try to understand these things with the help of some example very simple example right. So, now suppose I simply want to write down a function for computing a

function like  $y$  equal to  $f$   $x$  equal to  $x$  square; that means, I simply give the input and it gives me the output as an  $x$  square.

So, for that I give this  $y$  and here a name  $abc$ , right and after that I write down here function and inside the parenthesis I write down here  $x$  which is my input variable then after that I write this curly bracket and within this curly brackets I write down the syntax for computing the  $x$  square which is  $x^2$  very simple and after that if I try to enter the this will create a program whose name is here  $abc$  and the R console.

Now, the next question comes here that if you want to see what is this program or what are the contents of this program you can simply type  $abc$  or the function name on the R console. So, you can see here I have in this screenshot I have given the here the program after that I want to see how this  $abc$  looks like. So, it will give me here this type of outcome. So, that will give us what are the contents of this function.

And after that if you want to execute it then this is the way. Write down the name of the function and inside the parenthesis try to give the input values. If there is only one value then you have to give only here one value, but if there are more than one values then you have to give the input in the same order in which they have been defined in the function that is what you have to keep in mind, ok.

So, now if you try to see here when I try to write down here  $abc$  3 on the R console, it means  $x$  is equal to a 3 and when  $x$  is equal to here 3 what will happen here? I try to show you this  $x$  takes value here 3 this will now come inside the curly bracket and it will try to make it here 3 square and then whatever is the outcome that will be given here as a 9.

So, you can see here when you want to execute it, it is very simple just try to write down the name of the program and within the parenthesis you try to give all the values of the input variables in the same order in which you have given it in the program while defining the program while writing the program that is very important and if you try to interchange it then you will make a mistake, right.

For example, if your function you have written it is like here say  $x$ ,  $y$  and you try to give suppose here  $x$  equal to 2 and  $y$  to 3 then you have to write down here  $f$  say 2 comma 3,



but if you write like f 3 comma 2 then; obviously, x will become a 3 and y will become 2 and you may get a wrong outcome right. So, that you have to be careful.

Similarly, if you want to execute this function for x equal to 6. So, simply try to write down here abc inside the parenthesis 6 and then it will give you the value here 6 square and similarly if you want to execute here abc 9; that means, x equal to here 9 the outcome is going to be here 9 square which is 81. So, you can see that a if you want to deal with the single variables in the function it is very simple and now instead of x square you can write whatever you want to do, ok.

(Refer Slide Time: 14:14)

**Functions (Two variables)**

$z = x^2 + y^2 = f(x, y)$

```
> abc = function(x,y){
  x^2+y^2
}
```

$x=3, y=4$   
> abc(3, 4)  $3^2 + 4^2 = 9 + 16 = 25$   
[1] 25

$x=10, y=10$   
> abc(10, 10)  $10^2 + 10^2 = 100 + 100 = 200$   
[1] 200

$x=-2, y=-3$   
> abc(-2, -3)  $4 + 9 = 13$   
[1] 13

```
# Console
> abc = function(x,y){
+   x^2+y^2
+ }
> abc(3,4)
[1] 25
> abc(10, 10)
[1] 200
> abc(-2, -3)
[1] 13
>
```

Now, let me try to give you an example here where I try to take two variables. Suppose I want to have here a function like as here z is equal to x square plus y square, right. So, this I can write down here say f of x, y like this. So, I try to give it here a name say abc yeah I am trying to take the same name because for the sake of convenience is otherwise you can take a different name also. So, it is here function f u n c t i o n and then within the parenthesis in inside this argument I try to write the input variables.

So, there are two input variables x and y. So, I write them say x comma y and after this within this curly brackets I try to write down the commands here what I want to compute. So, I want to compute here x square plus y square. So, I try to give it here x hat

2 plus y hat 2 right and now in case if you try to execute it on the R console what you have to do?

You simply have to write down here abc and if you want to execute it for x equal to 3 and y equal to 4, then you have to write down here abc within the parenthesis 3 comma 4. So, this will become here 3 square plus 4 square which is 9 plus 16 equal to 25. And similarly if you want to execute this program for x equal to 10 and y equal to 10 then you have to write down abc 10 comma 10 and it will give you the value here say 10 square plus 10 square which is here 200.

And similarly if you want to do for x is equal to minus 2 and y is equal to minus 3, then this value will become here 4 plus 9 is equal to 13 which is given here. So, you can see that it is not a very difficult job to execute this functions in the R console and this is here the screenshot also, right.

(Refer Slide Time: 16:08)

**Functions- Another example**

$$f(x) = x + \sin^2(x) + \cos^2(x)$$

```
> abc = function(x) {  
  sin(x)^2 + cos(x)^2 + x  
}
```

$x=9$   
> abc(9)  
[1] 10

$x=99$   
> abc(99)  
[1] 100

$x=-15$   
> abc(-15)  
[1] -14

```
R Console  
> abc = function(x) {  
+   sin(x)^2 + cos(x)^2 + x  
+ }  
> abc(9)  
[1] 10  
> abc(99)  
[1] 100  
>  
> abc(-15)  
[1] -14  
> |
```

Similarly, if I try to give you here one more example right suppose I want to write down the function here as say here f of x is equal to x plus sin square x plus cos square x, right well those who are from mathematics background can know that sin square x plus cos square x is equal to 1, but it is like there may be some candidates who may not have the mathematics background. So, that is why I am taking this example here ok.

So, now in this case for what I will do? I will simply try to give here the name of the function abc once again I am trying to take the same name just for the sake of convenience, but it is up to you what name you want to give then I write down here function and then inside the parenthesis the input variable here x there is only one variable and then I try to use my built in function for computing the sin and cos function. So, this becomes a sin of x hat 2 plus cos of x hat 2 plus x and then I try to write down these thing inside the this curly, brackets.

So, you can see here when I try to execute it on the R console and I try to give here x equal to here 9 what will happen here? Sin square 9 plus cos squared 9 that will become here 1 and 1 plus here 9 that will become here 10. Now in case if you try to take here x equal to 99 then what will happen here? That sin square 99 plus cos square 99 will become 1 and then 1 plus 99 this will become here 100.

And similarly, if you try to take here abc minus 15; that means, you want to execute the function at x equal to minus 15 and this will give you the value here say sin square x plus cos square x 1 minus 15 and this will give you a the value here minus 14.

(Refer Slide Time: 18:00)

**Functions- Another example**

Calling a function without an Argument  $1^3 \ 2^3 \ 3^3$

```
abc = function()
  for(i in 1:3)
    print(i^3)
}

> abc()
[1] 1
[1] 8
[1] 27
```

$i=1, 1^3$   
 $i=2, 2^3$   
 $i=3, 3^3$

```
> abc = function() {
+   for(i in 1:3) {
+     print(i^3)
+   }
+ }
> abc()
[1] 1
[1] 8
[1] 27
> |
```

11

So, you can see here it is not a very difficult thing to do it on the R a console also, but sole let me try to give you here some more examples so, that you can be confident that when you are trying to do it on the R console. So, let me try to take here one more

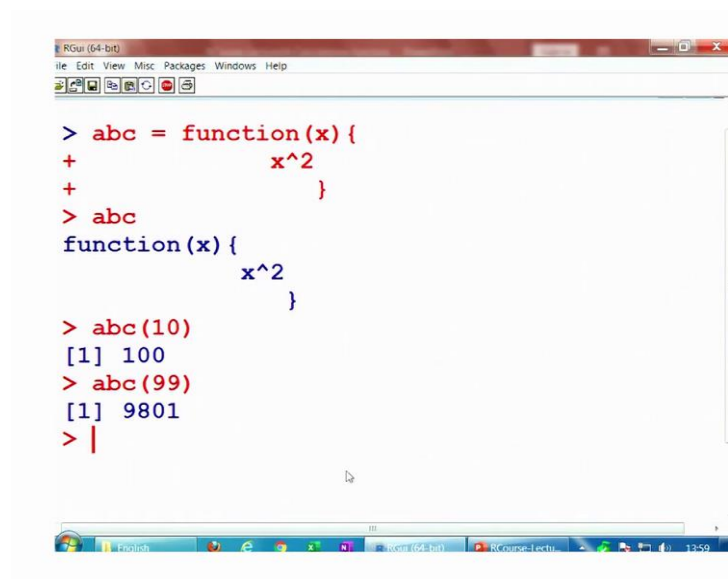
function where I am not trying to give any input variable right and simply I want to print say 1 cube, 2 cube and 3 cube their values. So, I try to write down here the function abc and then I try to give here the function and now there is no input because everything I am trying to give it inside the command.

So, I will simply leave it here blank then I try to write down here this curly bracket number 1 and I try to close it here and then I try to use here a loop say for i in 1 2 and 3 this will execute whatever I am trying to write down under this curly bracket number 2 which is here print i cube that is all. So, you know now you can understand these things very easily. So, what will happen here?

This i will come to here 1 and then it will try to print here 1 cube then i will come to 2 it will print here 2 cube, then i will come to here 3 and it will try to print here 3 cube. So, this will be here 1 8 27 and in order to execute it you simply have to give here abc and just the parenthesis you do not have to write any value here, right.

Now, we try to first see that how these functions are working in the R console they are very simple thing. So, and I am sure that now you have understood it and that was the reason that I wanted to first explain you before you try to do it so, that you do not get scared that what will happen when it is executed in the R console.

(Refer Slide Time: 19:41)

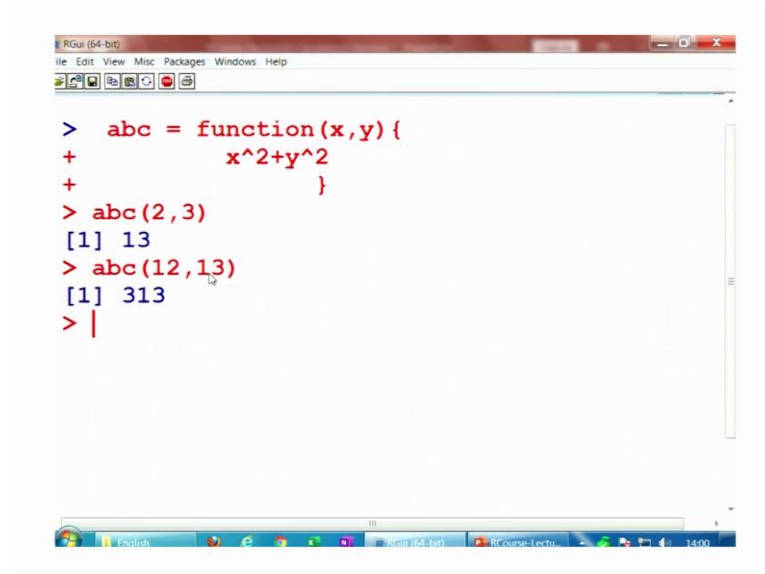


```
> abc = function(x) {
+       x^2
+     }
> abc
function(x) {
  x^2
}

> abc(10)
[1] 100
> abc(99)
[1] 9801
> |
```

So, you can see here this is my here function abc this is your function and if you want to execute it this will become here say abc 10 this will become here 100 if you want to execute here for x equal to say here 99 this will become here 9801 which is 99 square. And similarly if you try to take here this example that your function is x square plus y square.

(Refer Slide Time: 20:21)



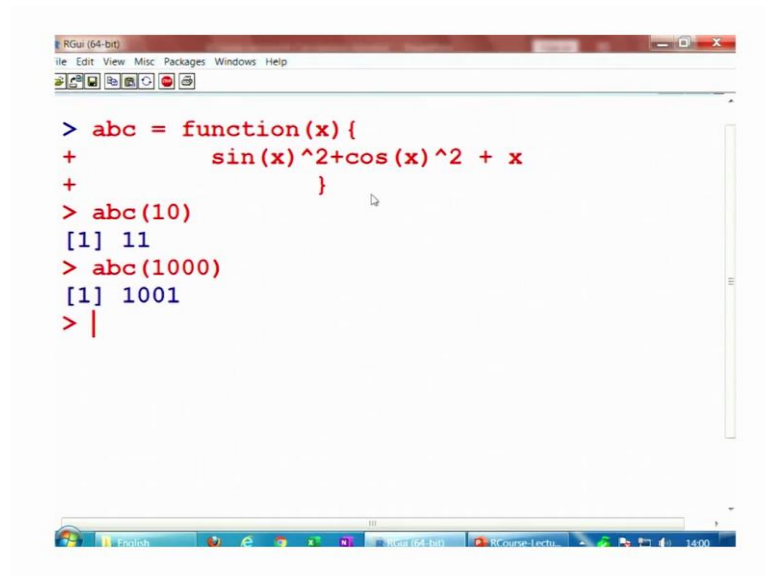
```
RGui (64-bit)
File Edit View Misc Packages Windows Help

> abc = function(x,y){
+       x^2+y^2
+     }
> abc(2,3)
[1] 13
> abc(12,13)
[1] 313
> |
```

So, once again if you try to see here I try to copy and paste this function to save some time, but now this has got 2 variables x and y. So, when you are trying to execute it you have to give here 2 values x equal to 2 and y equal to 3 in the same order which you have defined in the function.

And if you try to here enter it will give you 2 square plus 3 square which is 13 and similarly if you try to find out here the square of say 12 square plus 13 square you can give here x equal to 12 and y equal to 13 and it will give you the value 313 and so on. Now, similarly if you try to take here this example in which we are trying to compute the sin square x plus cos square x plus 1.

(Refer Slide Time: 21:04)

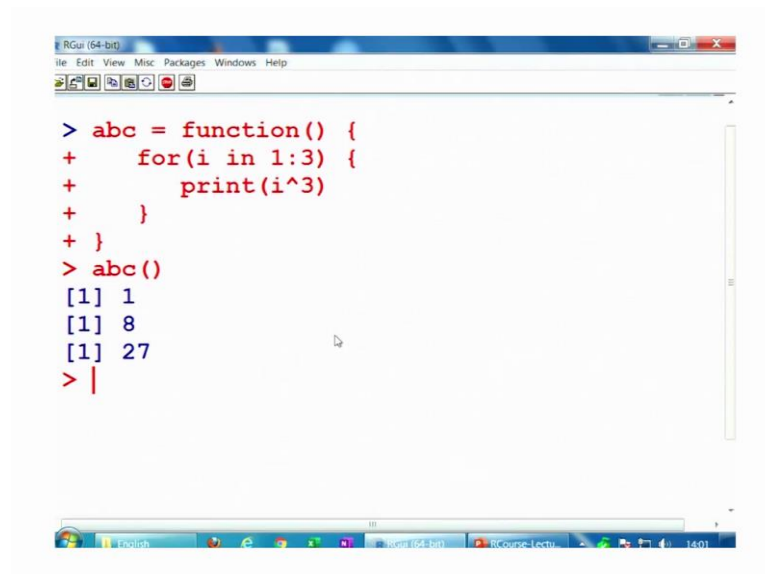


```
> abc = function(x) {
+   sin(x)^2+cos(x)^2 + x
+ }
> abc(10)
[1] 11
> abc(1000)
[1] 1001
> |
```

So, this is 1 plus x, but I am trying to write down the same function. So, you can see here if you want to execute here this will become here say abc 10 this will become here 1 plus 10 this is 11 because sin square 10 plus cos square 10 will be equal to 1 and similarly if you try to take it here abc 1000 at x equal to 1000 it will give you the same value a similar value like 1001.

So, you can see here that these functions are not difficult at all and if you try to execute here this function where you do not need to write anything.

(Refer Slide Time: 21:39)



```
> abc = function() {
+   for(i in 1:3) {
+     print(i^3)
+   }
+ }
> abc()
[1] 1
[1] 8
[1] 27
> |
```

So, here now you can see here say I am trying to print here 1 cube, 2 cube, 3 cube, but I do not give here now any value inside the parenthesis and if you try to enter here it will give you this value 1, 8, 27. So, now, we come to an end to this lecture and you can see here. This function concept is very simple to understand and in order to write down the function you need all sorts of information which we have learnt in the earlier lecture and what we are going to learn in the further lectures.

Function is the soul of the programming and besides these three very simple example now you can go back to your earlier two lectures where we had taken the examples on the function using for loop and while loop and try to now see that what we were trying to do that was not a very difficult thing.

So, now, I would say why do not you take very simple example do not take any complicated example at this stage after you practice after you learn more thing you will become a very good programmer and I am confident that you can write very complicated and longer programs also but at this moment if you do not know the programming then try to take very simple example and try to execute it try to use some built in functions also inside your programming.

And in case if you know the programming in other languages then also try to programs can be written in the R language also. So, you try to practice it and I will see you in the next lecture till then goodbye.