

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

Lecture – 21
Vector Indexing

Welcome to the next lecture on the course introduction to R software.

You may kindly recall that in the earlier lecture we started a discussion on vector indexing and we learn different ways of manipulating the index of a vector which is giving us an information about the position of an element in the vector and using those index values and using the logical operators we had created different types of data sets.

Now, we will continue on the same topic of indexing a vector or how to play with the index of a vector and we try to learn some more elementary concepts. So, now, here I am trying to now deal with the a string vector.

(Refer Slide Time: 01:12)

String vector
The elements of a vector can be named. ←
Using these **names**, we can access the vector elements.

names is used for functions to get or set the names of an object

```
> z <- list(a1 = 1, a2 = "c", a3 = 1:3)
> z
$a1
[1] 1
$a2
[1] "c"
$a3
[1] 1 2 3
```

names(z)
[1] "a1" "a2" "a3"

Handwritten annotations: Arrows point from 'a1', 'a2', 'a3' in the code to their respective elements in the output. A note says '@structure: call 1'.

What is a string vector? The string is a something like an alphabets characters. So, now, I am trying to play with the index of a vector whose entries are something like alphabets, but now you have to see I have used a wrong terminology usually when we call vectors or matrix their entries are in general some numbers.

Now, I am saying characters. So, we had d1 a couple of lectures back there these things can be d1 using the list command, in the list we can have all sorts of data objects character numbers mattresses and anything. So, now, I am trying to introduce here of command what is called here as names and using this names we can access the vector elements for example, suppose in a class there are 3 students, sitting at first position, second position and third position and suppose their names are student 1, student 2 and student 3.

Now, how our teacher will call them? They will simply say student number 2 please come, what is really happening if you try to analyze we are simply calling a student by his name and the name of the student to whom I call is student 2 this is a similar concept in R programming also that whenever we are trying to arrange these characters or say numbers or say anything in a particular order. Then these things can be arranged through the command list and we want to call a particular element or a particular value by its name inside the program

So, the question is how to do it, that is what we are going to learn in this lecture by taking certain examples. So, first of all what we have to understand that all the elements of a vector they can be named, what does this mean you may recall that when you went to your schools and colleges your parents had given you a name, but inside your college you are given a new name. What was the new name? Your roll number and as soon as you say your roll number your unique identity is established.

So, similarly whenever we are trying to deal with the vector elements all the elements can be given a unique name and now using those names we can access say any element of that vector. So, how to do it? Let us try to see here suppose I try to create here a list since this is the list. So, it can have each and everything numbers or say alphabets or anything else.

So, I try to take here 3 elements, element number 1, element number 2 and element number here 3. So, you can see here the first value here is a a1 equal to 1; that means, the value here is 1, but I am trying to give it a name a 1 and I would call this value a 1 from here 1 the second value is a character. Here c and I am giving this c a name say a 2 and my third entry is, this is a sequence of numbers 1, 2 and 3 and this I am giving a name a 3 and this will create a list and if I try to enter here the values of here z I get here this

thing. This is my first value, this is my second value and this is my here third value and these values are characterized by the names, this is the name of first value this is the name of second value and this is the name of third value.

Now, the question comes why this name is important? Why cannot we call a value by its numerical value or say its face value. Now please try to look at this example very carefully suppose my objective is call 1, now you can see here there are 2 places where we have 1, 1 place is here, here we have 1 and this is another place where we have 1. So, now, the confusion is which 1 do I call or out of these 2 which is the 1 to whom I am calling, this situation is very similar to a situation inside the class where there are 2 students with the same name and when the teachers calls them by their name both the student comes, but the teacher wants any particular student to come.

So, that is why they have been given our names which is their roll number and just by calling the roll number say roll number 5 and roll number 10 they may have the same names, but I am trying to identify them by a different name and when I say roll number 5 only the student with roll number 5 will come, that is the concept here which we are trying to discuss ok.

Now, once I have created this list I would like to know what are their names or in some other language I can say suppose you have got a list and you want to know what are the names which are given to all the elements, then I can use here a command names and inside the bracket I will try to write down the variable name which can be anything here in our case this is here z. So, and we get here an output like this one a1, a2 and a 3 and these a1, a2 and a3 these are the same thing same name which we have given here, here and say here right let us try to do it on the R console and try to see what do we obtain here.

(Refer Slide Time: 09:06)

```
□ String vector
Suppose want to change just the name of the third element.
> z <- list(a1 = 1, a2 = "c", a3 = 1:3)
> names(z)[3] <- "c2"
> z
$a1
[1] 1
$a2
[1] "c"
$c2 a3
[1] 1 2 3
```

So, you see here I have created the list same list with the names a1, a2 and a3 right and suppose I want to know what are the names in this list z I simply you have to type here names and I get here the outcome (Refer Tim: 09:36) let us try to see what happens next and before that I have given here a screenshot of whatever we have done.

Now, suppose I have created this list and also I have given the names, now the question is this suppose I want to change the name of someone, then how to get it done and is it really possible I am said this yes this is possible, but what is the command for that thing. Suppose in the list that we have just created called as here say z which is given here this is my here name first a2 is my name 2 and a3 is my here name 3 and suppose I want to change the name of third element a3 then how to get it done.

So, the syntax theories that you try to call a particular name by the command n a m e s all in a small letters inside the brackets you try to write down the name of the variable that is containing the list and then use square brackets and try to write down here the position, the address of the element which you want to call and here I am try to write down here 3; that means, I want to call the third element and now I am trying to give the third element a new name say c2 and after that if you try to see the value of here z.

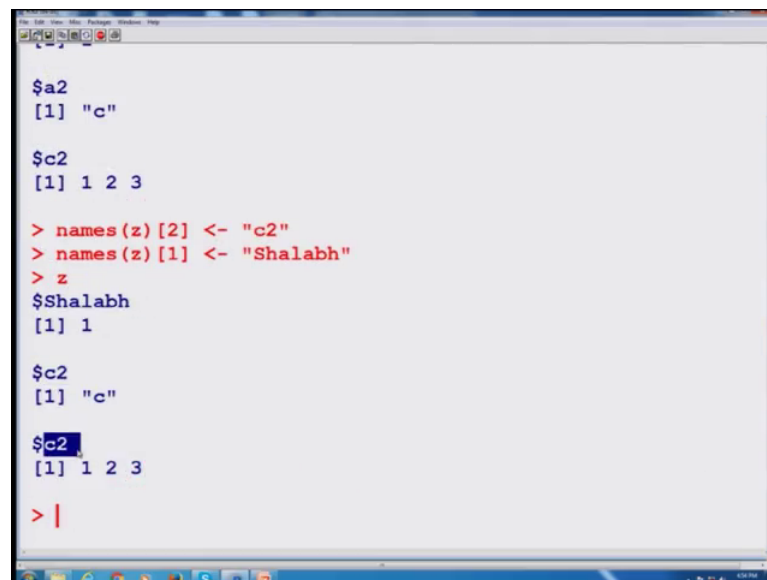
The first name remains as as such, second name remains as s such a1 and a2, but the third name which was earlier a3 this is now here change as c here c2, but one thing what you have to keep in mind the values are not changing I am changing only the name, that

is something like inside the class if a teacher announces that from tomorrow the student whose name is today say this ram will be called a Shyam from tomorrow, but the person will remain the same.

The marks obtained by is a ram or a Shyam they will remain the same. So, the same thing is happening here if you try to see here this output or the values they are the same as which are given over here 1, 2, 3.

Let us try to do it over the R console and c what do we get here.

(Refer Slide Time: 12:51)



```
File Edit View Misc Package Windows Help
$ a2
[1] "c"

$ c2
[1] 1 2 3

> names(z)[2] <- "c2"
> names(z)[1] <- "Shalabh"
> z
$Shalabh
[1] 1

$ c2
[1] "c"

$ c2
[1] 1 2 3

> |
```

So, I am trying to create here this list you can see here it has got 3 names and now I am trying to change here the name here c2 names here c2.

Now, if I try to write down here what is the value of here z you can see here that earlier it was a 3 which I have highlighted on the screen and now it becomes c2 which I have highlighted on the screen and suppose I want that the second name may also be changed and suppose I say the name of second and third position value should be the same. So, I can do here the same thing that I can now change the value at the second position and call it here say c 2 and suppose I also want to change the name of the first position and let me call it say I want to keep the name Shalabh my name. Now if you try to see what you get when you press here z first name becomes Shalabh my name, second name c2 and third name c2 right.

Now, let us come back to our slides and now in the next slide I have given the screenshot of the outcome that you can see what here now I try to take here 1 more example and try to see what we learn from here, suppose I try to create here a vector by here c not by list remember.

(Refer Slide Time: 14:58)

```
String vector
Example
names is used for functions to get or set the names of an object
> x <- c(water=1, juice=2, lemonade=3)
> names(x)
[1] "water" "juice" "lemonade"
> x["juice"]
juice
  2
```

Handwritten notes:

- I know the vector
- I know the name
- I do not know the value of the name.

I try to give it here 3 values say 1 2 and see here 3 and I give this a name, I give water for value 1, the name juice for value 2 and say lemonade for the value 3.

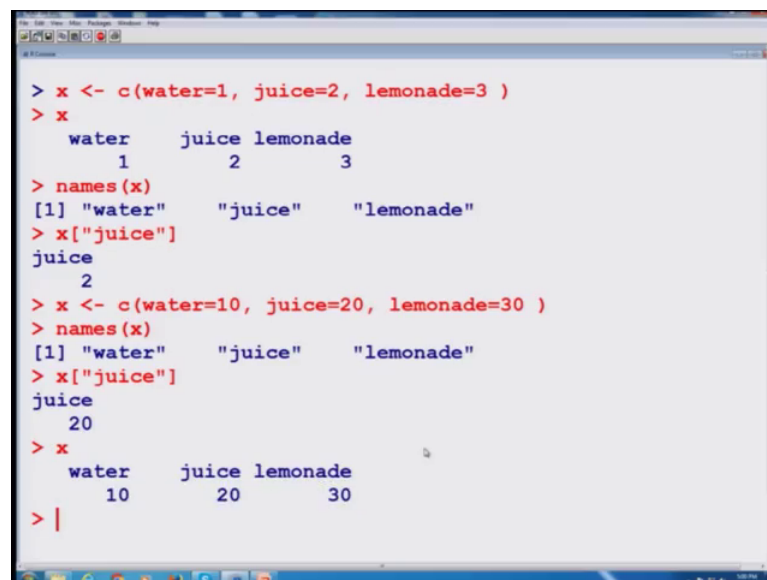
Now, suppose somebody has given me the vector x and if I want to know what are the names given to each of the element. So, I try to use here the same command here names of x and as soon as you press enter you will get here the same outcome water juice and lemonade. This is the name of the first position value uses the name of the second position value which is at the second position also and lemonade is at the third position and this is the name of the third value right.

Before we try to do the same example over the R console, let us try to see what do we get here. Now, suppose I try to take another question suppose someone has given me a vector and I know that there is some name as juice, but I want to know what is the value of this name juice. So, what I try to do here is the following. So, first my question is this, I know the vector, I know the name, what I do not know is this I do not know the value of the name value of a given name.

Suppose I want to know what is the value which the name juice is taking. So, I can write down here as a follows write down the variable name see here x and sin this is a name this is not a number. So, I try to write down the name inside this double quotes and I try to write down here the name juice and as soon as you enter it will let me know the name is juice and its value here is 2.

Let us try to do the same thing over the R console and see what do we get here.

(Refer Slide Time: 17:50)



```
> x <- c(water=1, juice=2, lemonade=3 )
> x
  water    juice lemonade
     1         2         3
> names(x)
[1] "water"  "juice"  "lemonade"
> x["juice"]
juice
 2
> x <- c(water=10, juice=20, lemonade=30 )
> names(x)
[1] "water"  "juice"  "lemonade"
> x["juice"]
juice
 20
> x
  water    juice lemonade
     10         20         30
> |
```

So, you can see here this is my here vector x containing 3 names water, juice and lemonade and their value is 1, 2 and 3 right and suppose I do the following suppose I try to find out their names. Names of x this comes here only water juice and lemonade and suppose I want to find out what is the value of the name juice in the vector x this comes out to be 2.

Suppose I try to give it here a different name for example, I can say thirty lemonade to be 30, juice to be 20 and water to be 10 and then let us try to see what happens you can see here names is here water, juice lemonade same, but what is the value of here juice this comes out to here 20, because if you try to see here the x vector is now here water juice and lemonade, but it is 10, 20 and 30 right.

Now, let me try to do here something more.

(Refer Slide Time: 19:20)

```
> x
  water  juice lemonade
    10    20     30
> x <- c(juice=10, water=20, lemonade=30 )
> x
  juice  water lemonade
    10    20     30
> x["juice"]
juice
  10
> |
```


Suppose the quizzes your here x and I try to change the value of this name, suppose I give it my 30 and suppose I change it. I try to interchange the values of here, juice with water and value 10 with here juice right. So, you can see here now this x becomes like this and now if I try to find out the value of juice this comes out to be here 10.

So, these are the different manipulations that you can do with the indexing. So, now, let us come back to the slide and here is the outcome or the screenshot of the outcome.

(Refer Slide Time: 20:26)

```
Empty index
> x <- 1:10
>x
[1] 1 2 3 4 5 6 7 8 9 10
> x[]
[1] 1 2 3 4 5 6 7 8 9 10
```

x
x[] → no value
Empty index



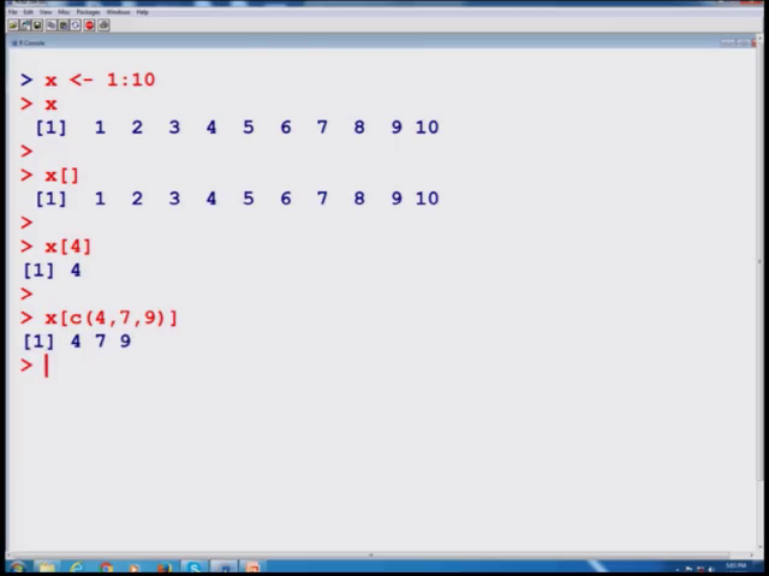
```
> x <- 1:10
> x
[1] 1 2 3 4 5 6 7 8 9 10
>
> x[]
[1] 1 2 3 4 5 6 7 8 9 10
```


Now, one thing what you have to just understand that when we do not write anything inside the index then what happens; that means, when an index is empty.

Let us try to see here suppose I try to generate a sequence of here 1 to 10 and that is assigned to a variable x. So, this is my vector values 1, 2, 3, 4 up to 10. So, I have 2 options to get these values either I simply write here x or I try to write down here x and inside this bracket there is no index no value, but this is empty index.

So, even then I will get the same values over here and this is the screenshot but let us try to see what happens on the R console. So, if I say here x 1, 2 here 10.

(Refer Slide Time: 21:30)



```
> x <- 1:10
> x
[1] 1 2 3 4 5 6 7 8 9 10
>
> x[]
[1] 1 2 3 4 5 6 7 8 9 10
>
> x[4]
[1] 4
>
> x[c(4,7,9)]
[1] 4 7 9
> |
```

You can see here x comes out to be here like this, but if I try to write down x in the format of an index, but I am not giving here any index, the entire sequence comes up, but in case if I try to give it here any value say here 4 then you will see only the fourth value comes or if I try to do here something like combine 4 value and say seventh value and say ninth value then you see here that 3 values comes over here.

So, this is another thing which you have to keep in mind and now let us come back to our slides and try to understand something more.

(Refer Slide Time: 22:20)

Matrices created from Lists
List can be heterogeneous (mixed modes).
We can start with a heterogeneous list,
give it dimensions, and
thus create a heterogeneous matrix
that is a mixture of numeric and character data:

Example

```
> ab <- list(1, 2, 3, "X", "Y", "Z") → 6 values
> dim(ab) <- c(2,3) → # of columns
> print(ab)
```

	[,1]	[,2]	[,3]
[1,]	1	3	"Y"
[2,]	2	"X"	"Z"

Handwritten notes: "number" above 1, 2, 3; "character" above "X", "Y", "Z".

Here now I am going to be take a little bit diversion from whatever I was doing, but it is related to lists as well as with c index. That we have understood that list can be heterogeneous, what do you mean by heterogeneous? That means, the entries inside the list can be of different modes there can be a mixed modes right. So, the question is that suppose I have got a list and suppose I want to create a matrix you have to keep in mind 1 thing that I am not talking of a matrix that contains only the numerical values, but matrix here is something where I can enter the values which have got a particular and unique address.

So, I want to create a matrix where there can be anything, some characters or say numbers. So, the question is once I have got a list can I create a matrix answer is yes, but how? So, the steps are as follows we can start with any heterogeneous list and then I have to mention the dimension of the matrix and this will create a heterogeneous matrix and in case my list has mixed modes then this matrix is going to be a mixture of numeric and character data, can I try to take an example and try to understand what I am trying to say here.

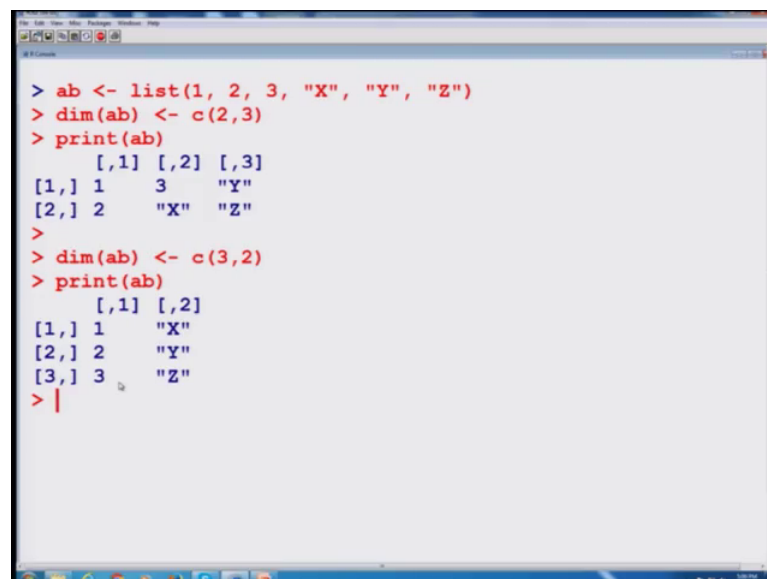
Suppose I try to create here a list by the name a b and this list have 2 types of entries, 1 is numbers and another type is say character. So, I am taking her 3 numbers 1, 2, 3 and 3 characters x y z and this is the list. So, altogether there are 6 values, 3 numbers and 3

characters, now I try to define what is the dimension of the matrix which I want, since there are six values the dimension of the matrix can be 2 by 3 or say 3 by 2.

So, suppose I try to define the dimension of the matrix as see here 2 and 3 and they are combined at the operator c. So, this 2 is trying to define the number of rows and this 3 is trying to define the number of columns. So, now, I am saying here that this list is now converted to something whose dimension are 2 by 3 and now if I try to see what is the structure of a b. So, I try to print my a b and we get here this type of thing and you can see here that these values are arranged according to the values in the list from 1 2 then 3, then here 4 and then here 5 and then here sixth value to the 1, 2, 3 and the fourth, fifth, sixth values which are here x y and z.

Let us try to do it over the R console and see what do we get over here suppose I try to create this list.

(Refer Slide Time: 26:31)



```
> ab <- list(1, 2, 3, "X", "Y", "Z")
> dim(ab) <- c(2,3)
> print(ab)
      [,1] [,2] [,3]
[1,] 1    3   "Y"
[2,] 2    "X"  "Z"
>
> dim(ab) <- c(3,2)
> print(ab)
      [,1] [,2]
[1,] 1    "X"
[2,] 2    "Y"
[3,] 3    "Z"
> |
```

And then I try to give its dimension and then I try to print here a b you can see here this comes out to be like this.

Now, suppose I change the dimension of maybe a say 3 by 2 instead of 2 by 3. So, now, if you try to print here a b once again, you can see here that the values are arranged in the matrix of 3 by 2 right and the next slide is the screenshot of the outcome.

So, now we stop here with this vector indexing and list and on the next turn we will try to take up some other issues my request to all of you is that you try to take some example and try to practice it over the R console. Unless and until you type this commands yourself over the R console, try to play with the R software and try to see that how the outcomes are coming and are they really matching with what you thought.

This will really enrich your knowledge in order to write a good program we have to understand what computer is doing, how are computed things and once the thinking process or the thinking steps of computer and our match we can do a good programming and whatever we have done up to now these are very small ingredients of programming, this is just like if you want to cooks good food there are different types of spices. So, you can cook a good food only when once you know that taste of each and every spice and that will give you a good dish similar is with the programming also. So, you learn how to cook a good dish till then goodbye.