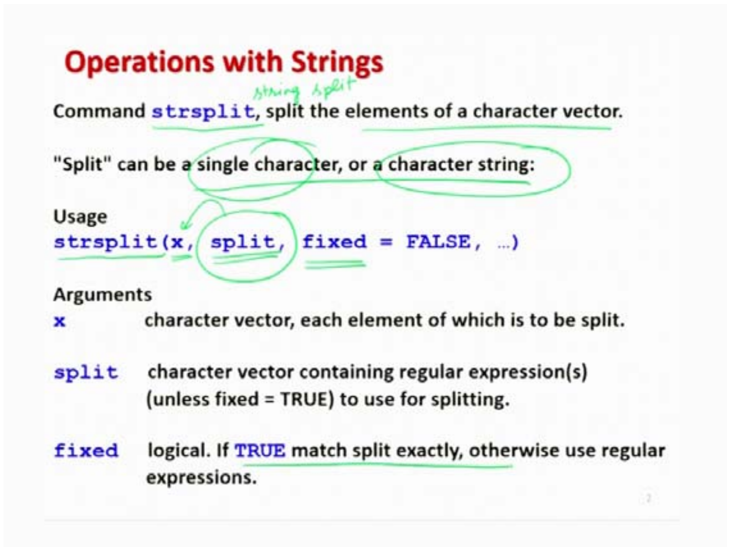
Foundations of R Software
Prof. Shalabh
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Strings - Display and Formatting
Lecture - 38
String splitting

Hello friends. Welcome to the course Foundations of R Software. You can recall that in the last lecture we had talked about that how you can paste different types of strings. So, now, in this lecture we will continue on a similar topic, but we will try to learn here how you can split the string. Well, this is a very common operation that sometimes you need in the data manipulations and database at some particular values which you want to define.

So, how to do it in the R software this is the topic what we are going to learn in this lecture. And for that I will try to take a couple of examples and through those examples I will try to show you that how R can also be used to split the strings at some desired points. So, let us begin our lecture.

(Refer Slide Time: 01:08)



Operations with Strings

Command $\overset{\text{string split}}{\texttt{strsplit}}$, split the elements of a character vector.

"Split" can be a single character, or a character string:

Usage
```
strsplit(x, split, fixed = FALSE, …)
```

Arguments

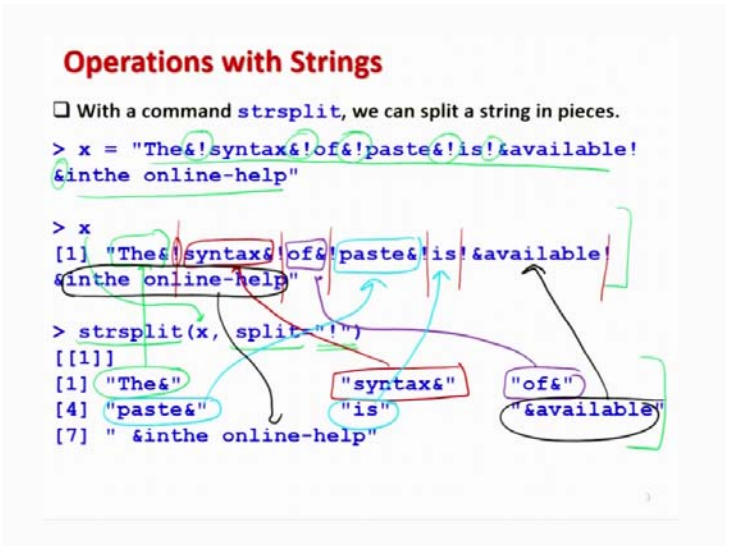
x	character vector, each element of which is to be split.
split	character vector containing regular expression(s) (unless fixed = TRUE) to use for splitting.
fixed	logical. If TRUE match split exactly, otherwise use regular expressions.

So, in case if you want to split the elements of a character vector, the command is strsplit which means string split, right. Now, this split can be a single character, that can also be a character string, right. So, because I will try to explain you in this lecture through some example that what do I mean by saying that the split can be a single character or character string. So, the command here is like this, that you write down here strsplit all lower case alphabet and then you have to give the string or the character vector which you want to split.

And then after that you have to give here an option here split and this is character vector that contains an expression and there you want to inform the R software that at a this is the point which is given under the split. There the string under the x has it to be splitted. And then you have here one more command here fixed that can take two possible values that is true or false.

So, if it is true then it matches the split exactly, otherwise it will use the usual regular expressions, right. So, I will try to show you with some example, then all these statements will become very clear.

(Refer Slide Time: 02:16)



Operations with Strings

- With a command strsplit, we can split a string in pieces.

```
> x = "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> x
[1] "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> strsplit(x, split="!")
[[1]]
[1] "The&"                    "syntax&"      "of&"
[4] "paste&"                  "is"           "&available"
[7] " &inthe online-help"
```

So, now suppose I try to take a statement here like this, the syntax of paste is available in the online help and you can see here I am trying to give here certain characters and an exclamatory signs at these places, right. So, this is your here statement and you want to suppose split this statement x, at all those places where you have an exclamatory sign.

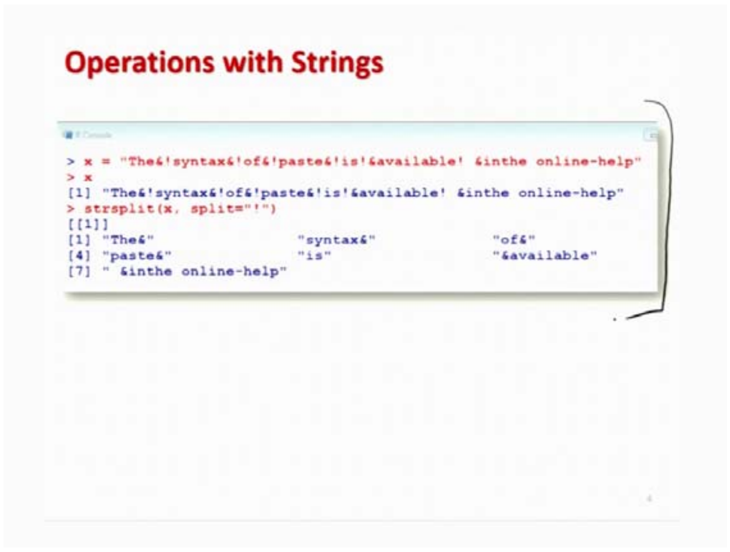
So, this will be splitted here, splitted here, splitted here, splitted here, here and then finally, means say here like this.

So, now, how to get it done? So, this is very simple in R that you can do such an operation. You simply write down here strsplit and then you try to write down this expression here, as I told you. And then you try to use here the option split equal to and now within double quotes try to give the character at which you want to split. So, you can see here as soon as you execute it you will get here this type of outcome.

And if you try to understand this outcome, you can see here this outcome this is the split of this thing you can see here. And similarly, if you try to take here this outcome this syntax and, this is here the 2nd split, right. And similarly, if you try to take here the 3rd one off & and this is here the 3rd split. And similarly, if you try to take here the 4th one here this is a paste & and. So, this is your here this paste&.

And then you have here this is also here and after that you have here the and available. So, this is here this one and then after this you have here in the online help like this here. So, this is coming here. So, you can see here this is what is happening, that this string split command is simply trying to split the statement at those places where you have given the instruction under the option split, right.

(Refer Slide Time: 04:10)

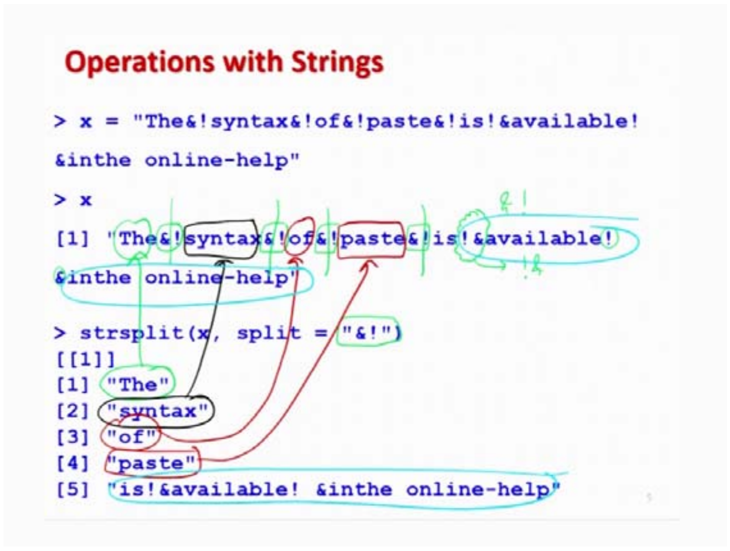


Operations with Strings

```
> x = "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> strsplit(x, split="!")
[[1]]
[1] "The&"                    "syntax&"                 "of&"
[4] "paste&"                  "is"                      "&available"
[7] " &inthe online-help"
```

And this is here the screenshot of the same operation.

(Refer Slide Time: 04:13)



Operations with Strings

```
> x = "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"
> x
[1] "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> strsplit(x, split = "&!")
[[1]]
[1] "The"
[2] "syntax"
[3] "of"
[4] "paste"
[5] "is!&available! &inthe online-help"
```

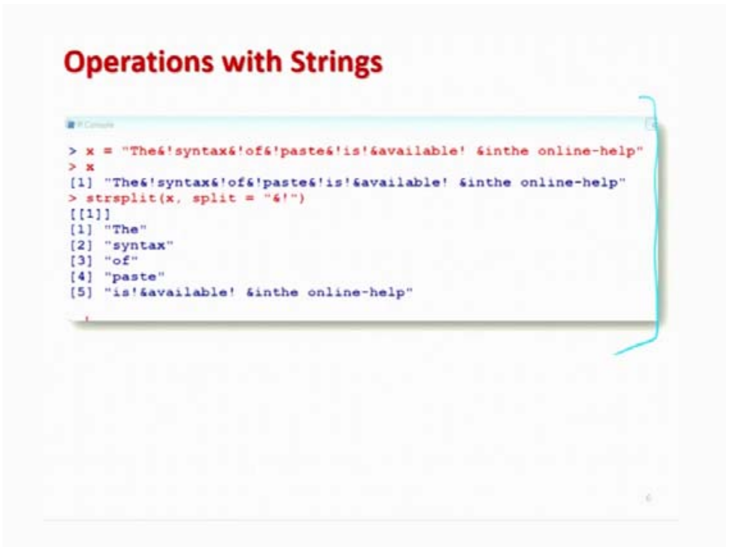
Now, I try to give you here this one more example and this is the same statement what I just explained you, but now I am trying to change my character at which I want to split. So, first you try to see, I am trying to give here now the split here as say like and& exclamatory sign within the double quotes, earlier I had given only the exclamatory sign.

So, if you try to see what will happen wherever is the exclamatory sign this is going to be splitted. So, if you try to look at the outcome and the statement what you have given you can understand it very easily and after that if you try to see the next symbol is coming here. And after this the next symbol is here, after that here is the next symbol. What do you think about this? This is not the symbol; why? Because you have given here and & exclamatory sign whereas, this is here exclamatory sign and &.

So, it will not be splitted here and after this if you try to see in continuation this is here exclamatory sign and here and. So, it is not going to be splitted here also. So, this split is going to happen here at the then syntax, then it off then here paste and after that the entire string will come at one place. So, if you try to see here the first outcome here is corresponding to this the because the split is happening here and then here it is here syntax, which is happening here and then if you try to see here the third here is off which is happening here.

And then after that you have here paste. So, paste is happening here and after that there is no more splitting and hold this statement this and here this is coming at one place. So, you can see here this is how the things are happening, in this when we try to split the string and this is here the screenshot of the same outcome, which I just shown you here, right.

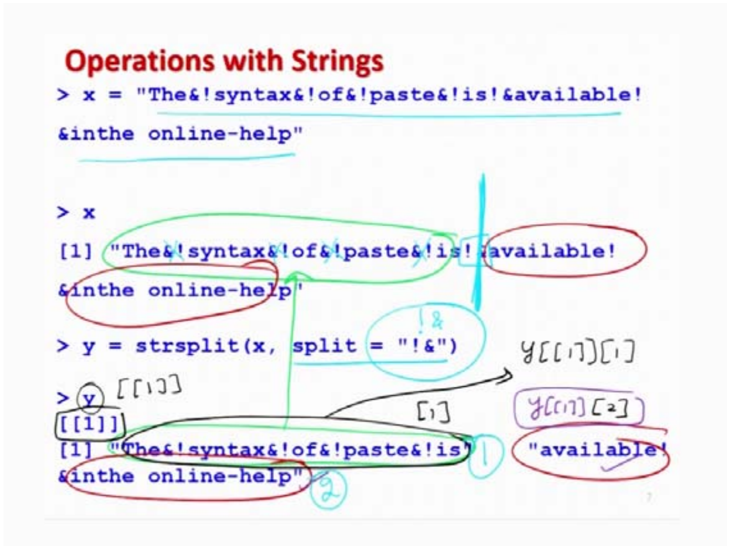
(Refer Slide Time: 05:50)



Operations with Strings

```
> x = "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> strsplit(x, split = "&!")
[[1]]
[1] "The"
[2] "syntax"
[3] "of"
[4] "paste"
[5] "is!&available! &inthe online-help"
```

(Refer Slide Time: 05:56)



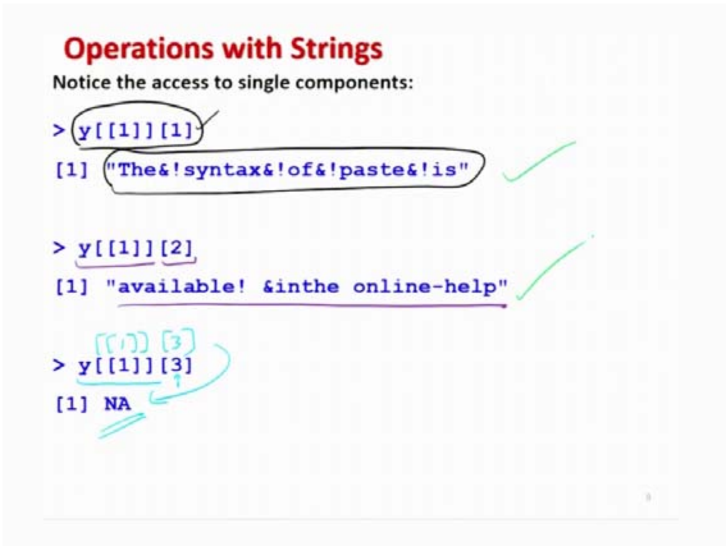
Operations with Strings

```
> x = "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> x
[1] "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> y = strsplit(x, split = "!&")

> y
[[1]]
[1] "The&!syntax&!of&!paste&!is" "available!
&inthe online-help"
```

So, you can see here. So, now similarly if you try to now replace your here this spit command here by this and & exclamatory sign. Now, you can see what will happen, this exclamatory sign and & where is this happening. If you try to see this is here and nowhere else, if you try to see you cannot consider this, you cannot consider this, you cannot consider this, you cannot even consider this because they are and & exclamatory sign, right. So, what will happen here? That if you try to take the same contents here and if you try to split at this symbol here.

Then it is going to be broken only at one place here and you can see here very easily that this is the first split which is happening here with this part and the second part which is here like this and here this, which is happening here this, right. So, you just have to just follow my this pen, that how I am doing it, but you can see here this is how the string split work.

(Refer Slide Time: 06:52)



Operations with Strings

Notice the access to single components:

```
> y[[1]][1]
[1] "The&!syntax&!of&!paste&!is"

> y[[1]][2]
[1] "available! &inthe online-help"

> y[[1]][3]
[1] NA
```

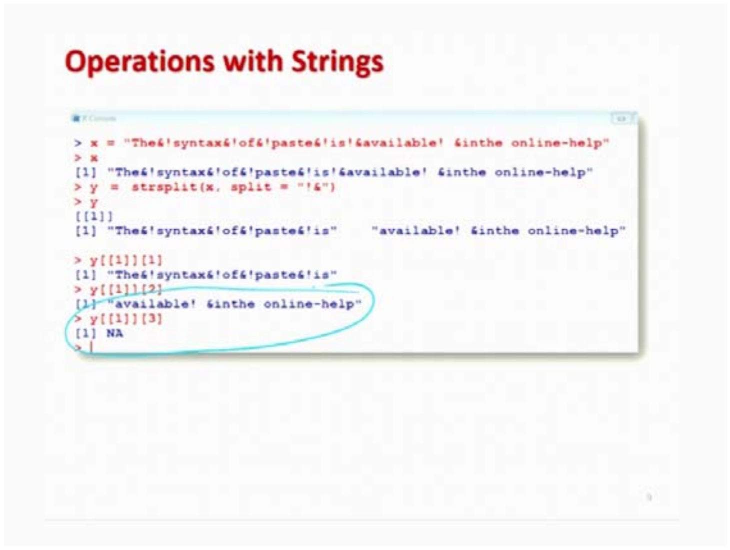
So, now in this case if you want to access any particular component how are you going to work on this? Now, actually you have in different types of data objects. So, similarly if you try to look here, that in this case you have to observe here like this one this is inside double square bracket it is here one and after this is the value which is here, this is here bracket number 1, square bracket number 1 and the second value is here bracket number 2, square bracket.

So, if you want to suppose call the first element here what you have to write suppose you have stored this outcome in the vector y, you have to write down here the complete address, then first you write the one in the double square bracket and then you try to write down the location of your element which you want to call, right.

So, this is exactly what I have done here. That if you try to write down here like this that y and then first you have to write down the double square brackets and then you try to write down here this 1. So, this is calling this part this is the first part of your split and similarly if you want to call the second part of the split. How you can do it? You simply have to write this 1 and here 2, and if you try to see here this is like this available in the online help.

And if you try to verify it here, this is like this y and then you are simply trying to write down here 1 and then it is here 2 and this 2nd value will appear here.

(Refer Slide Time: 08:18)



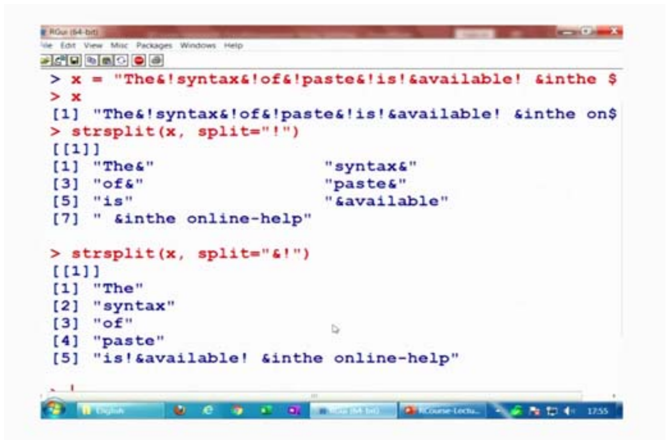
Operations with Strings

```
> x = "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> y  =  strsplit(x, split = "!&")
> y
[[1]]
[1] "The&!syntax&!of&!paste&!is"    "available! &inthe online-help"

> y[[1]][1]
[1] "The&!syntax&!of&!paste&!is"
> y[[1]][2]
[1] "available! &inthe online-help"
> y[[1]][3]
[1] NA
> |
```

And this is here the screenshot of the same operation. Now, in case if you try to see, what will really happen if you try to write down here y 1 and then after you write 3. So, in the double square brackets you are write here 1 and then here 3 and then you get here the outcome like edge here NA.

Why this is NA? Because this is not available, if you try to see here there are only two partition, this is the partition number 1 and this is here the partition number 2. So, there is no 3rd partitioning. So, it is giving you here an A, right and this is the same outcome which you can see here in this screenshot. So, let us try to first see these operations on the R console and then I will try to give you some more aspects.

(Refer Slide Time: 09:04)

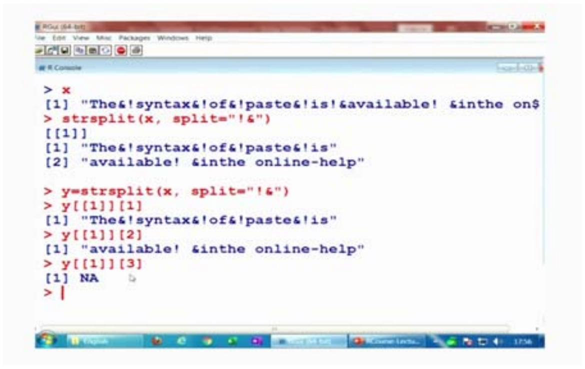


```
> x = "The&!syntax&!of&!paste&!is!&available! &inthe $
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe on$
> strsplit(x, split="!")
[[1]]
[1] "The&"                    "syntax&"
[3] "of&"                     "paste&"
[5] "is"                      "&available"
[7] " &inthe online-help"

> strsplit(x, split="&!")
[[1]]
[1] "The"
[2] "syntax"
[3] "of"
[4] "paste"
[5] "is!&available! &inthe online-help"
```

So, first let me try to write down this here x and you can see here. So, you can see here x is here like this. Now, if you try to write down here the command for this here string is split and you are trying to write down here like this split is equal to exclamatory sign. So, wherever is your exclamatory sign, this is partitioning here, right.

And now in this case if you try to write down the command here that split is equal to and exclamatory sign, you can see here, this is here this is here and so on. So, you can see here the change in the outcome. You can see here there is a difference here, now you can see here means earlier there were seven partitions now you have only here five partitions.

(Refer Slide Time: 09:44)

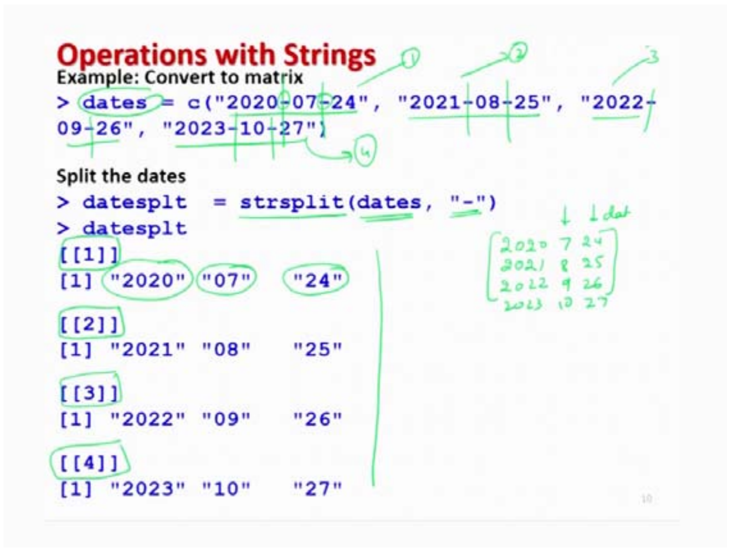


```
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe on$
> strsplit(x, split="!&")
[[1]]
[1] "The&!syntax&!of&!paste&!is"
[2] "available! &inthe online-help"

> y=strsplit(x, split="!&")
> y[[1]][1]
[1] "The&!syntax&!of&!paste&!is"
> y[[1]][2]
[1] "available! &inthe online-help"
> y[[1]][3]
[1] NA
>
```

And similarly, if you try to do here like this, let me try to clear the screen and then if you try to see here this is your here x. And if you try to write down here exclamatory sign and and, right, then you know that what is going to happen there is going to be only here two outcome, right. So, similarly if you try to save this outcome in some here variable here y and you want to call the first element.

So, first you have to write down here this 1 and then you try to write down here this one, right. So, it will give you the first value and similarly if you try to write down here 2, it will give you here the second value, right and, if you try to give here y 3. So, you can see here there is no y 3. So, it will give you here the NA, right. So, this is how you can access any particular value after splitting means any partition after splitting, right.

(Refer Slide Time: 10:36)



Operations with Strings

Example: Convert to matrix

```
> dates = c("2020-07-24", "2021-08-25", "2022-
09-26", "2023-10-27")
```

Split the dates

```
> datesplt  = strsplit(dates, "-")
> datesplt
[[1]]
[1] "2020" "07"   "24"

[[2]]
[1] "2021" "08"   "25"

[[3]]
[1] "2022" "09"   "26"

[[4]]
[1] "2023" "10"   "27"
```

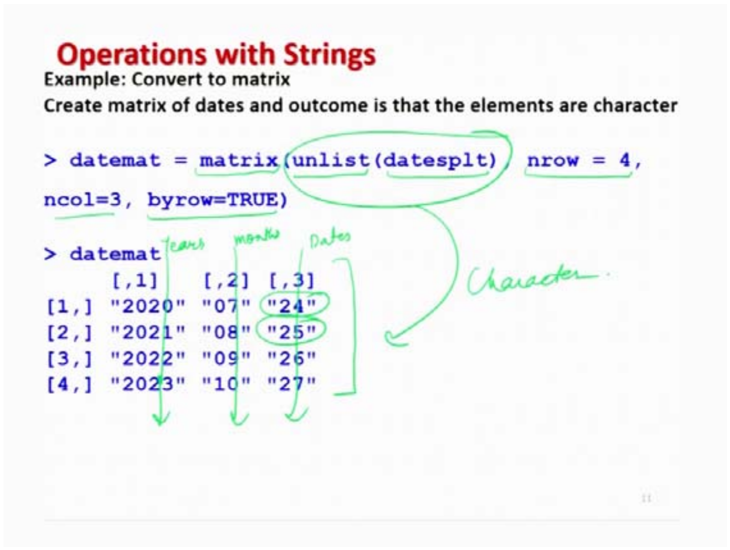
So, now we come back to our slides and now I try to give you here one more example. And this example is very interesting that you will see that I will try to split some dates to a matrix. This looks very strange how can you convert some dates here to a matrix. So, what I want here is that I have here this 4 dates, 24-7-2020, 24-July-2020, then 25th-August-2021 and then you have here 26-September-2022 and then 27th-October-2023.

And suppose I want to write down here a matrix like this one, that in the first column of the matrix I need here years like this then in the second column I want to write down here, the months like a 7, 8, 9 and here 10. And then I want to write down here the dates, like as here this is 24th-July 1st value, 2nd value 25th-August 3rd value is 26th-September and 4th value is 20 so here 7 October, right.

So, first I try to make a split of this data vector of four dates, at a point wherever I have this hyphen. So, you can see here this will be splitted here, then here, then here, then here, then here, here, right. So, for that I try to give here a statement strsplit, then here inside the parenthesis date and then within double quotes I try to give here the hyphen sign. So, you can see here now I have here this splitting, but you can see here the problem now is that you have here splitting, which is inside the double quotes.

So, these are essentially your characters number 1, number 2 you can see here that in the earlier examples you always took only one string. So, there was a partitioning and you were only observing there the this one inside the double brackets. But now here you can see here you have here say string number 1, string number 2 string number 3 and string number here 4. So, that is why you have here this four values in the square brackets, 1, 2, 3 and here 4. And after this whatever are the dates they are partition, right.

(Refer Slide Time: 12:39)



Operations with Strings

Example: Convert to matrix

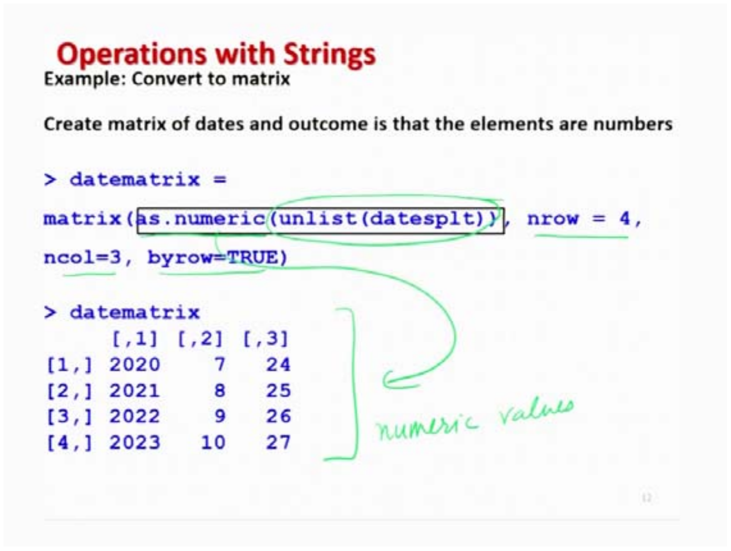
Create matrix of dates and outcome is that the elements are character

```
> datemat = matrix(unlist(datesplt), nrow = 4,
ncol=3, byrow=TRUE)

> datemat
     [,1]   [,2] [,3]
[1,] "2020" "07" "24"
[2,] "2021" "08" "25"
[3,] "2022" "09" "26"
[4,] "2023" "10" "27"
```

So, now what you can do? That now, if you try to see this outcome is going to be in the form of a list. So, first you would like to use the command here unlist, right. So, as soon as you try to here write down here unless, what will happen? The data which is stored in the date split, right, that will be free from this list. And then now you are trying to use here the command here matrix and then you try to use here nrow is equal to 4, ncol is equal to 3 and byrow is equal to TRUE.

And you can see here this data will be arranged here like this, right. So, this is your here 1st column of the year, 2nd column of the months and 3rd column of the date. So, this is your here year, then here months and then your here dates, right, but you have to observe here one thing, that once again inside this matrix all the values are in the form of a character only, they are inside the double quote, right.

(Refer Slide Time: 13:34)



Operations with Strings

Example: Convert to matrix

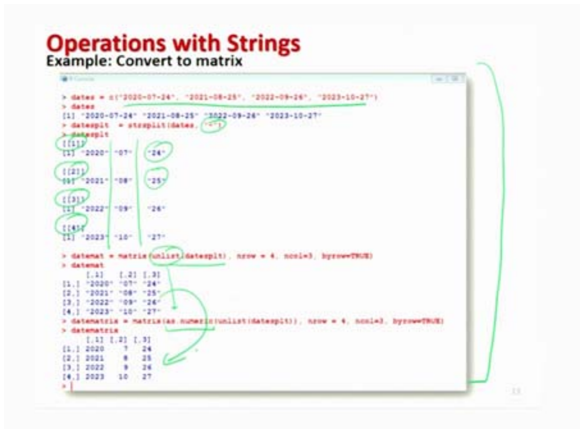
Create matrix of dates and outcome is that the elements are numbers

```
> datematrix =
matrix(as.numeric(unlist(datesplt)), nrow = 4,
ncol=3, byrow=TRUE)

> datematrix
     [,1] [,2] [,3]
[1,] 2020    7   24
[2,] 2021    8   25
[3,] 2022    9   26
[4,] 2023   10   27
```

So, now if you want to get rid of this character and you want to convert it into a number, because if you want to make any mathematical manipulations over the matrix, then all the elements inside the matrix have to be in the form of some number. So, now, you can use now here as new dot numeric and then you try to have this data that you have obtained through the unlist of datesplit data vector, right.

And again you are trying to use here nrow equal to 4 and ncol is equal to 3 and byrow is equal to TRUE and now you get here the same outcome you can see here. If you try to compare it here with this outcome so, when you are trying to use here only unless it was going to give you the data which is in the format of character, but if you try to use once again numeric over this data what you have obtained and the earlier outcome now this will give you here a numeric value. So, if you try to see here, the reason why I took this example, right.

(Refer Slide Time: 14:31)



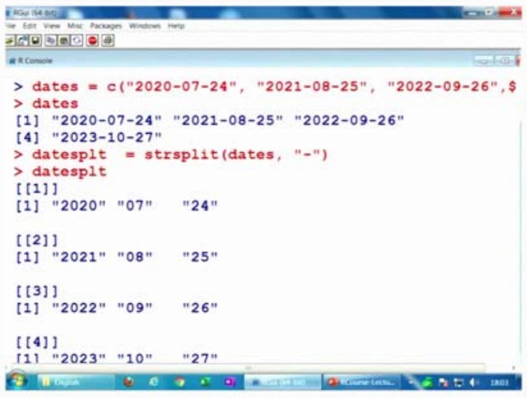
Operations with Strings
Example: Convert to matrix

```
> dates = c("2020-07-24", "2021-08-25", "2022-09-26", "2023-10-27")
> dates
[1] "2020-07-24" "2021-08-25" "2022-09-26" "2023-10-27"
> datesplit  = strsplit(dates, "-")
> datesplit
[[1]]
[1] "2020" "07"   "24"

[[2]]
[1] "2021" "08"   "25"

[[3]]
[1] "2022" "09"   "26"

[[4]]
[1] "2023" "10"   "27"

> datemat = matrix(unlist(datesplit), nrow = 4, ncol=3, byrow=TRUE)
> datemat
     [,1]   [,2] [,3]
[1,] "2020" "07" "24"
[2,] "2021" "08" "25"
[3,] "2022" "09" "26"
[4,] "2023" "10" "27"
> datematrix = matrix(as.numeric(unlist(datesplit)), nrow = 4, ncol=3, byrow=TRUE)
> datematrix
     [,1] [,2] [,3]
[1,] 2020    7   24
[2,] 2021    8   25
[3,] 2022    9   26
[4,] 2023   10   27
>
```

And you can see here this is the screenshot, right. So, in this screenshot you can see these are your here dates, when I try to split it they are getting here splitted, but everything is in the form of a list, you can see here these are the addresses, right. And so now, you wanted to get a rid of these addresses.

So, you first use the command unlist and then you use the command as dot numeric over this data what you have obtained here and you get here the mat, right. So, now let me first try to show you this outcome on the this screen, on the R console and then I try to show you here what is going to happen, right.

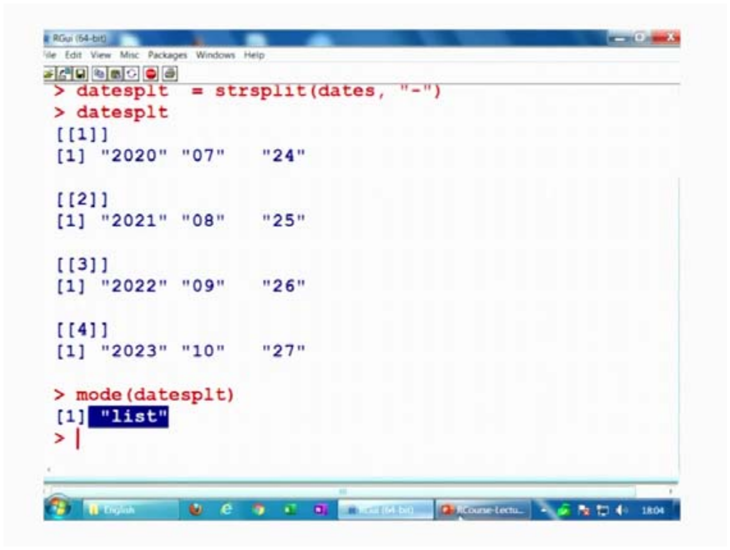
(Refer Slide Time: 15:05)



```
> dates = c("2020-07-24", "2021-08-25", "2022-09-26",$
> dates
[1] "2020-07-24" "2021-08-25" "2022-09-26"
[4] "2023-10-27"
> datesplt  = strsplit(dates, "-")
> datesplt
[[1]]
[1] "2020" "07"   "24"

[[2]]
[1] "2021" "08"   "25"

[[3]]
[1] "2022" "09"   "26"

[[4]]
[1] "2023" "10"   "27"
```

So, let me try to just copy this here data. So, first let me try to copy here these dates. So, these are your here dates. So, you can see here there are 4 dates, right. Now, you try to use here the command here, string with split over this data here and you try to store it in the value date split datesplt, you can see here like this, right.

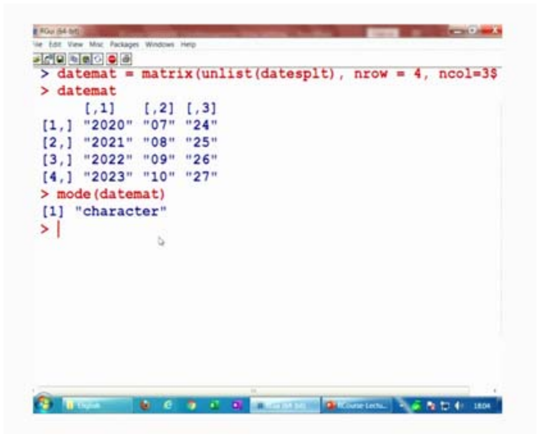
(Refer Slide Time: 15:30)



```
> datesplt  = strsplit(dates, "-")
> datesplt
[[1]]
[1] "2020" "07"   "24"

[[2]]
[1] "2021" "08"   "25"

[[3]]
[1] "2022" "09"   "26"

[[4]]
[1] "2023" "10"   "27"

> mode(datesplt)
[1] "list"
>
```

So, now if you want to see what is the mode of this datesplt, you can see here this is a "list" and that is why you want to first made this outcome independent of the features of the list. So, that is why I have to use here the command here unlist over this data vector and then I will try to show you that what is happening.

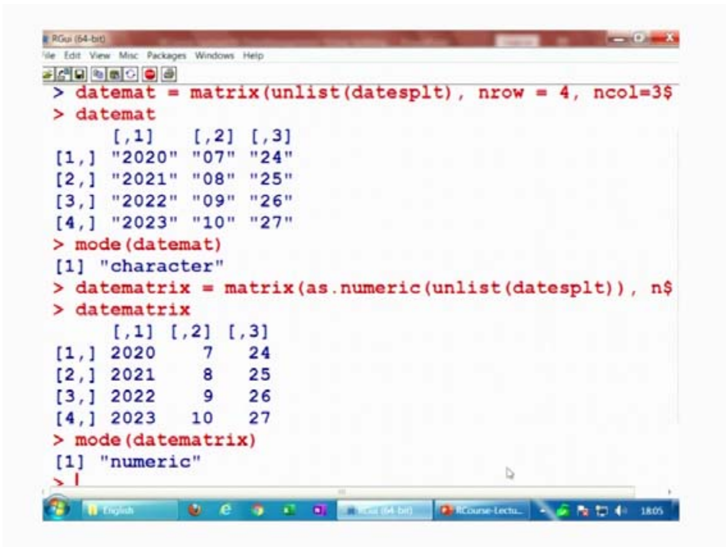
(Refer Slide Time: 15:55)



```
> datemat = matrix(unlist(datesplt), nrow = 4, ncol=3$
> datemat
     [,1]   [,2] [,3]
[1,] "2020" "07" "24"
[2,] "2021" "08" "25"
[3,] "2022" "09" "26"
[4,] "2023" "10" "27"
> mode(datemat)
[1] "character"
>
```

So, if you try to see here now what happens here, datemat. Now, you have used the command here unless on this earlier obtained data and you can see here that this is here. Now, like this, but this is in the form of character and if you try to find out the mode of this outcome here you can see here very clearly this is character. So, you want to get rid of this value here and you try to use here the command here as dot numeric.

(Refer Slide Time: 16:19)

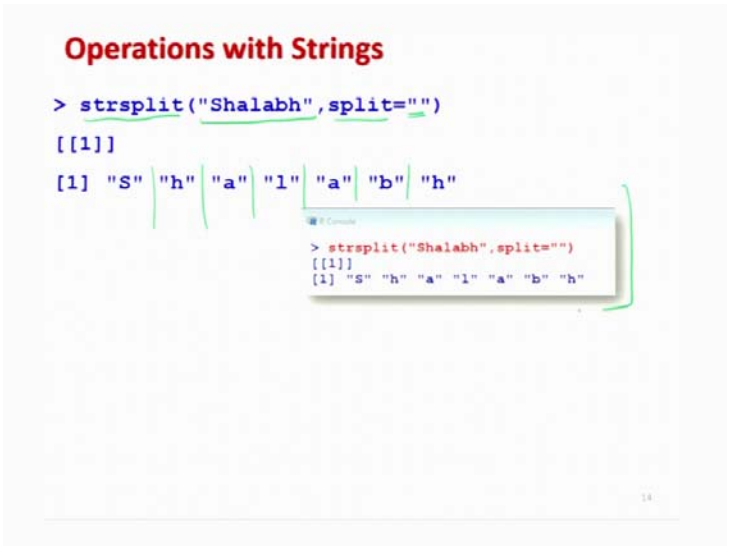


```
> datemat = matrix(unlist(datesplt), nrow = 4, ncol=3$
> datemat
     [,1]   [,2] [,3]
[1,] "2020" "07" "24"
[2,] "2021" "08" "25"
[3,] "2022" "09" "26"
[4,] "2023" "10" "27"
> mode(datemat)
[1] "character"
> datematrix = matrix(as.numeric(unlist(datesplt)), n$
> datematrix
     [,1] [,2] [,3]
[1,] 2020    7   24
[2,] 2021    8   25
[3,] 2022    9   26
[4,] 2023   10   27
> mode(datematrix)
[1] "numeric"
> 
```

And once you try to execute it here you can see here this date matrix comes here like this and if you try to find out here the mode of this date matrix, you can see here now this is numeric, right. So, if you want to do any mathematical operations over this matrix, then you can do it.

So, the reason why I took this example was that, if I ask you in the beginning of this example that if you have this type of dates and can you convert it into a matrix, possibly it will be difficult for you to think, but if you try to see whatever commands we have used here. If you simply try to use them intelligently in a logical way, possibly you can achieve and this is the best part of the programming that how you have to think the logic in which you can execute and can get what you want, right.

(Refer Slide Time: 17:07)



Operations with Strings

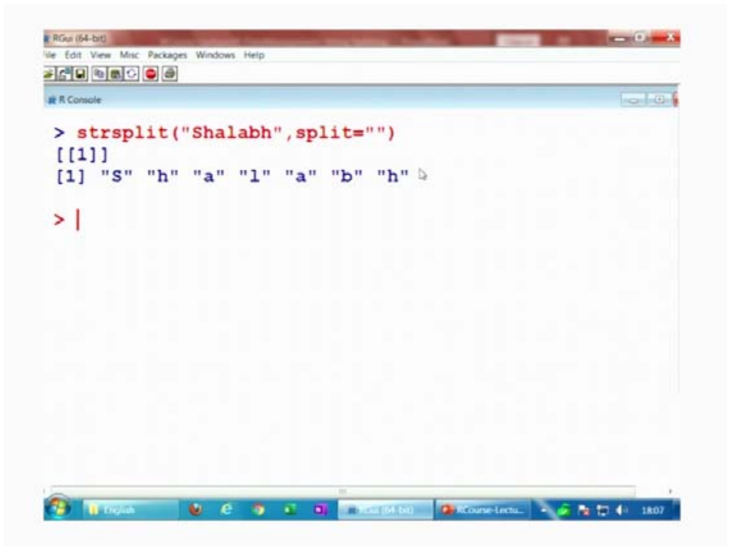
```
> strsplit("Shalabh",split="")
[[1]]
[1] "S" "h" "a" "l" "a" "b" "h"
```

```
> strsplit("Shalabh",split="")
[[1]]
[1] "S" "h" "a" "l" "a" "b" "h"
```

So, just as I shown you in the beginning, that when you are trying to use this string split then you can split it at strings as well as characters. So, up to now, whatever you have seen here you can see in this example you are trying to split these things as some strings, right. But now, I try to show you here that even if you want to split the individual characters, what you can say here you simply try to write down here string split and then in the split you do not write anything within double quotes you just do not say anything.

Just two double quotes that is all, right. For example, if I want to write down my name here, here say x say “s” “h” “a” “l” “a” “b” “h” which is a string here, but if you try to split it with this command you can see here this is splitted after every alphabet and this is here that is the screenshot. So, you can see here this is not a very difficult command and just try to show you here that how you can convert the individual string into strings as well as individual alphabets or individual character, right.

(Refer Slide Time: 17:59)



```
> strsplit("Shalabh",split="")
[[1]]
[1] "S" "h" "a" "l" "a" "b" "h"

> |
```

So, you can see here it is here like this ok. So, now, we come to and end to this lecture. So, this was a short lecture and I have just shown you here only one command that how you can split a string. But the example which I have shown you that will indicate you that why I have taken this command here and I have tried to show you that well in the beginning it looks that as if you are simply going to know how you are going to split the string that is correct.

But then how it is going to be used in various types of application that was not clear in the beginning. But now, at the end of the lecture you can see that how you have splitted the string and then you have converted the things into some numerical values. Similarly, you can think about such applications and then try to execute it.

Now, it depends on your capability that how you can think as you try to do it you will become a very good programmer. So, you try to practice it and try to become a wonderful programmer and I will see you in the next lecture. Till then, good bye.