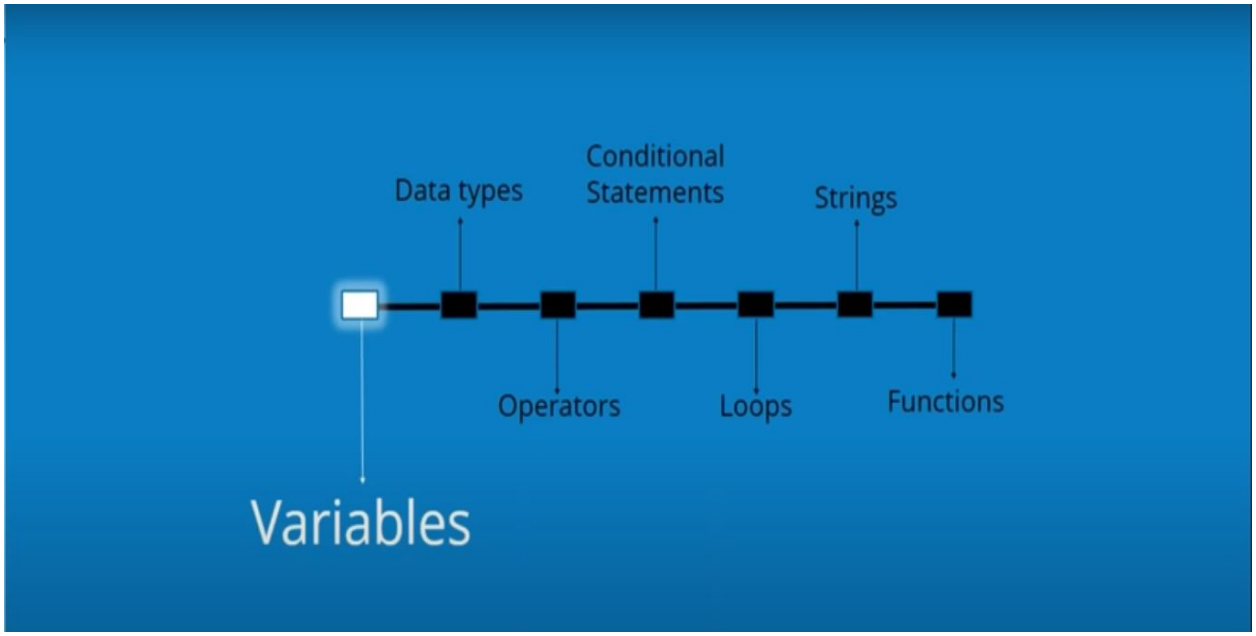
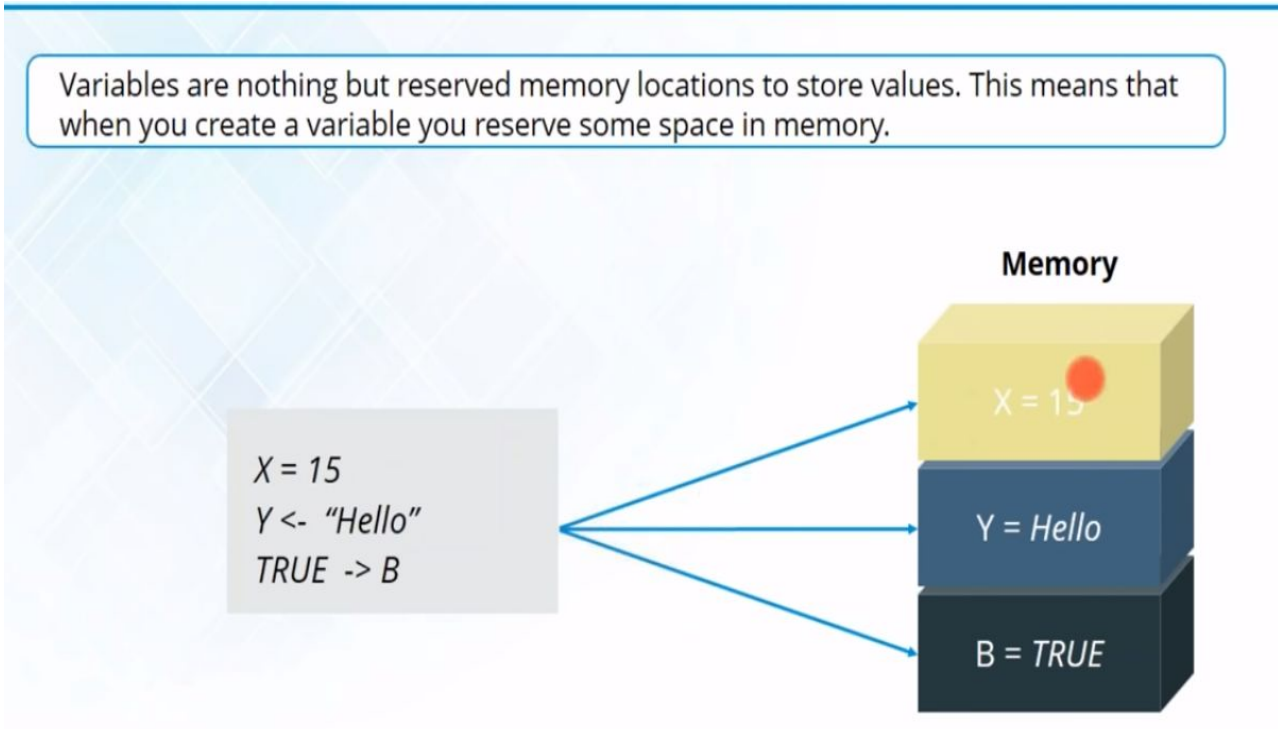

Data types
Conditional Statements
Strings
Operators
Loops
Functions
Variables

Variable :


Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.
Memory
X = 15
Y <- "Hello"
TRUE -> B
X = 15
Y = Hello
B = TRUE

Variable Name	Validity	Reason
var_name2.	valid	Has letters, numbers, dot and underscore
var_name%	Invalid	Has the character '%'. Only dot(.) and underscore allowed.
2var_name	invalid	Starts with a number
.var_name, var.name	valid	Can start with a dot(.) but the dot(.)should not be followed by a number.
.2var_name	invalid	The starting dot is followed by a number making it invalid.
_var_name	invalid	Starts with _ which is not valid

Deleting Variables

Variables can be deleted by using the rm() function. Below we delete the variable var.3. On printing the value of the variable error is thrown.

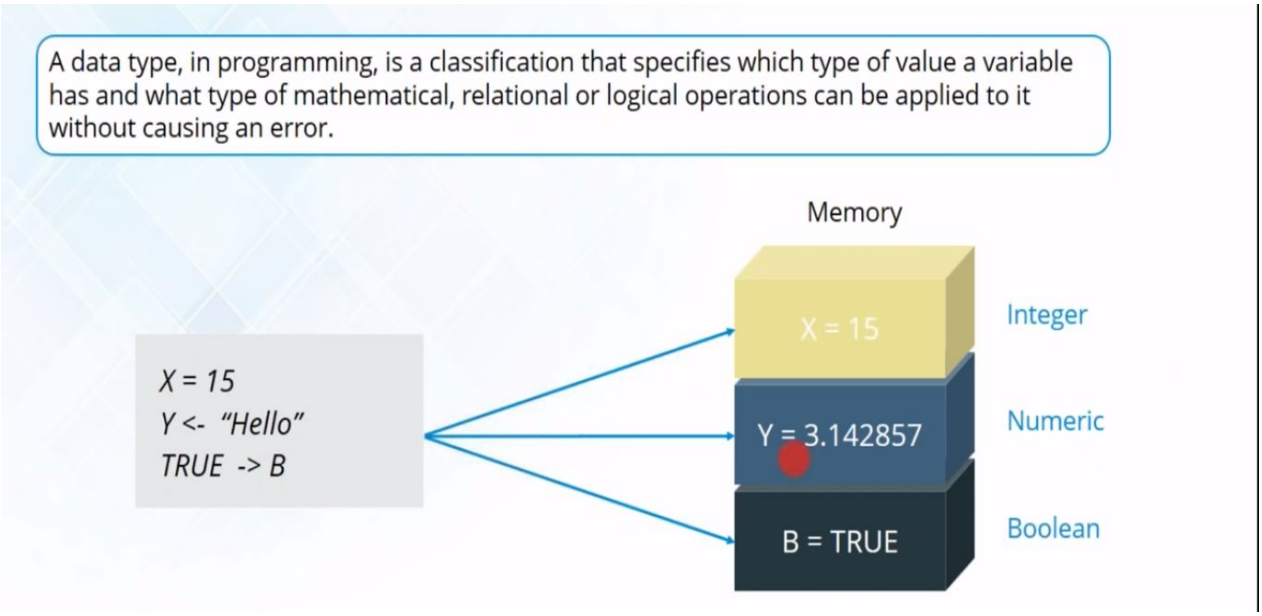
```
rm(var.3)
```

```
print(var.3)
```

When we execute the above code, it produces the following result −

```
[1] "var.3"
Error in print(var.3) : object 'var.3' not found
```

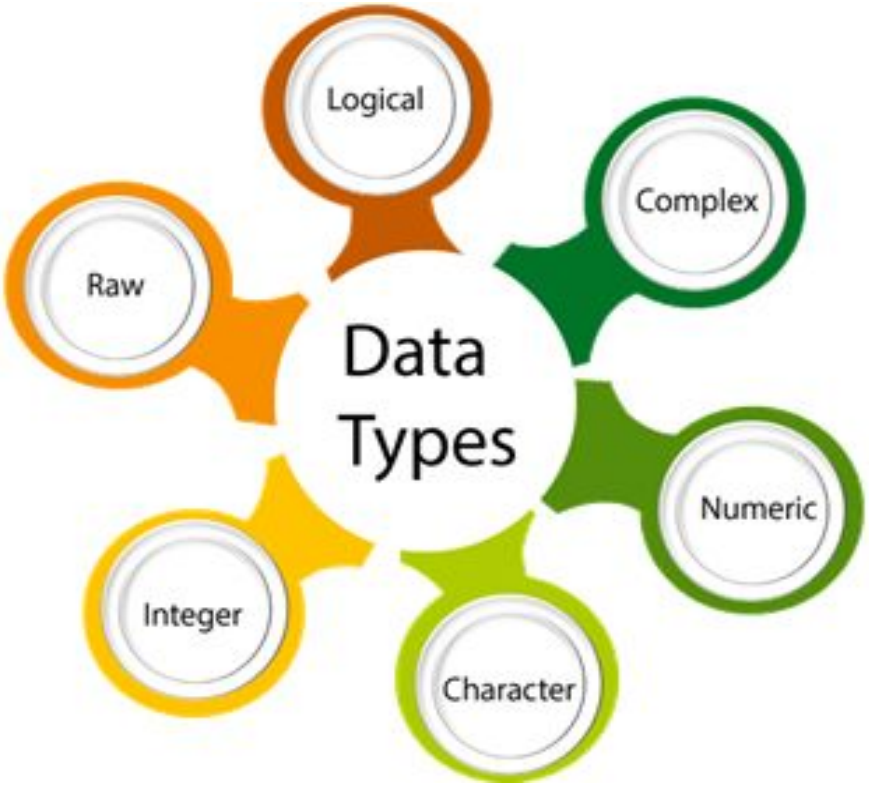
DataTypes in R-prog.



:

Data Types in R - Programming :

- **R's basic data types are character, numeric, integer, complex, and logical,Raw.**
- **R's basic data structures include the vector, list, matrix, data frame, and factors,array.**

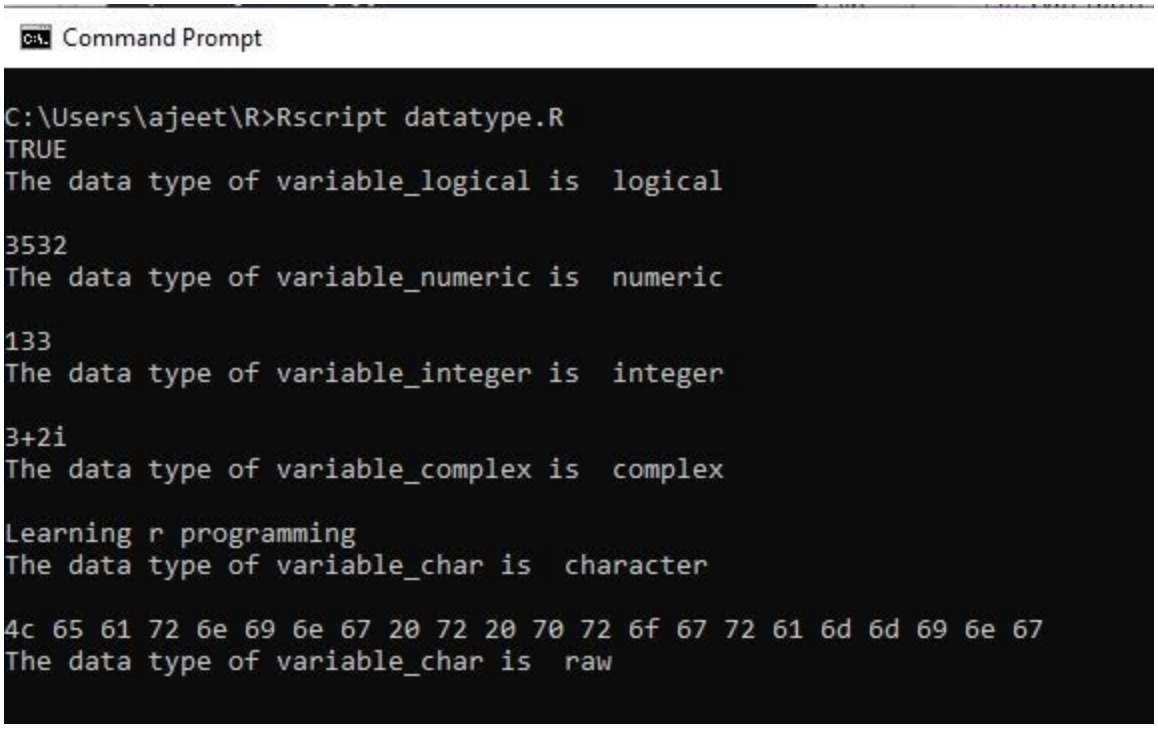



```
#Logical Data type
variable_logical<- TRUE
cat(variable_logical,"\n")
cat("The data type of variable_logical is
",class(variable_logical),"\n\n")

#Numeric Data type
variable_numeric<- 3532
cat(variable_numeric,"\n")
```

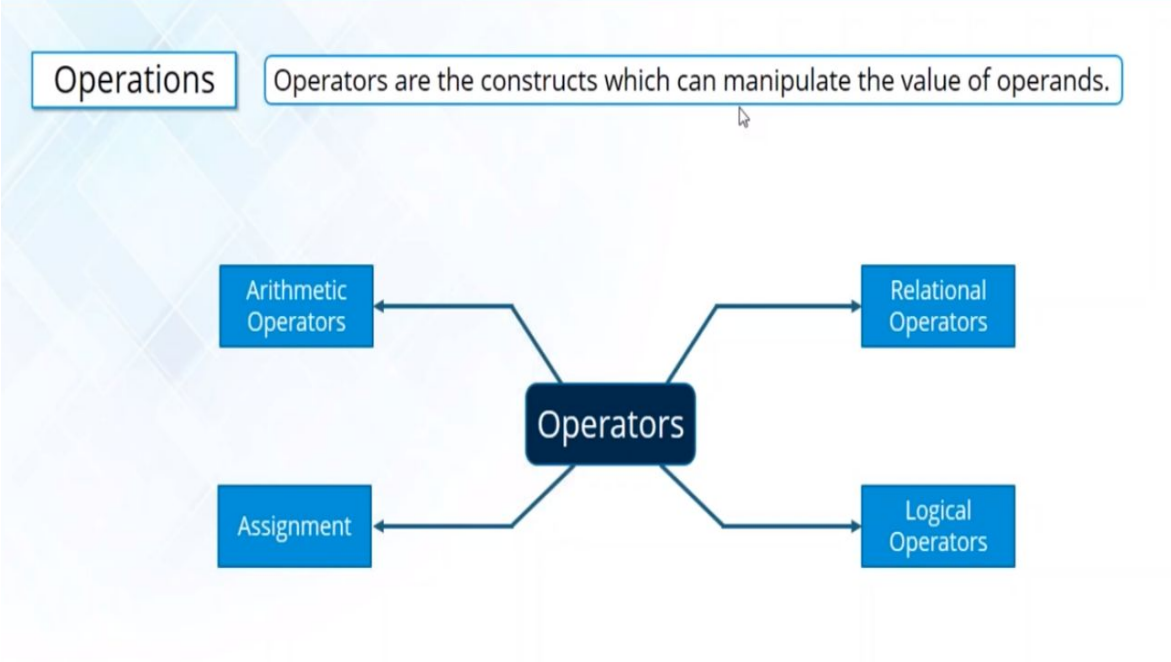
```
cat("The data type of variable_numeric is
  ",class(variable_numeric),"\n\n")

#Integer Data type
variable_integer<- 133L
cat(variable_integer,"\n")
cat("The data type of variable_integer is
  ",class(variable_integer),"\n\n")

#Complex Data type
variable_complex<- 3+2i
cat(variable_complex,"\n")
cat("The data type of variable_complex is
  ",class(variable_complex),"\n\n")

#Character Data type
variable_char<- "Learning r programming"
cat(variable_char,"\n")
cat("The data type of variable_char is
  ",class(variable_char),"\n\n")

#Raw Data type
variable_raw<- charToRaw("Learning r programming")
cat(variable_raw,"\n")
```

29. **cat("The data type of variable_char is ",class(variable_raw),"\n\n")**

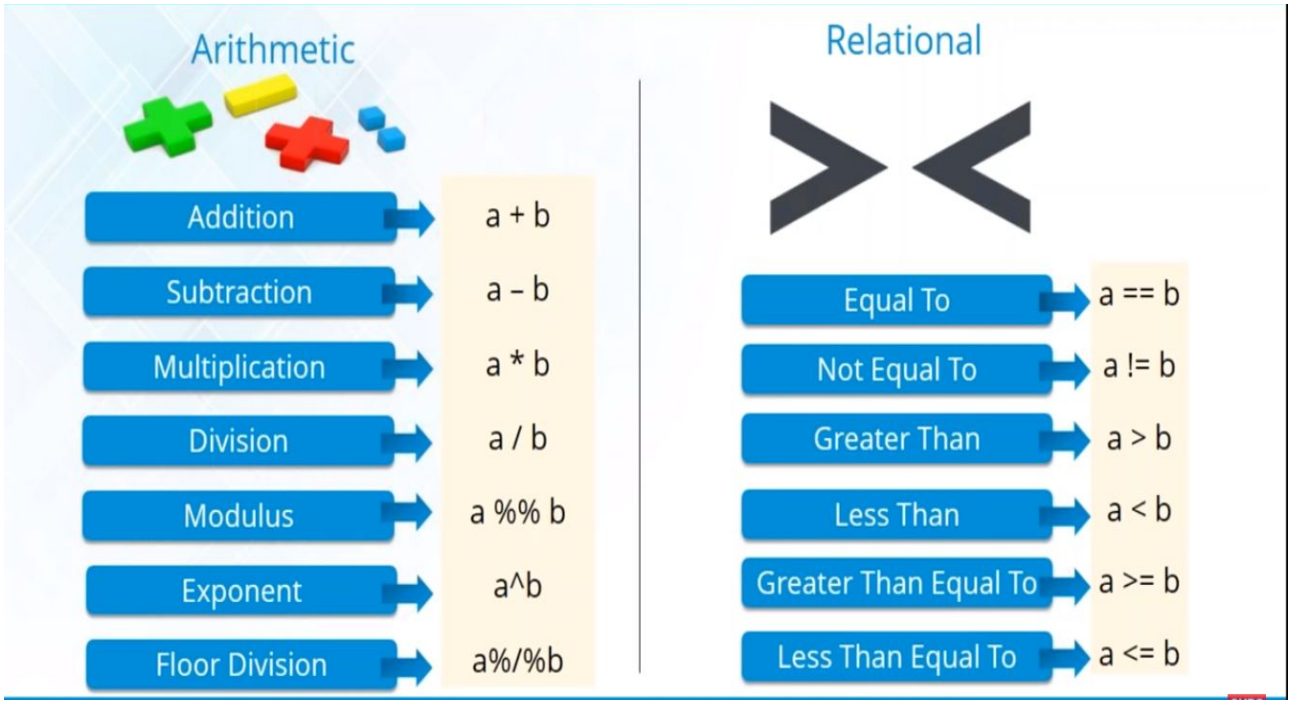
When we execute the following program, it will give us the following output:

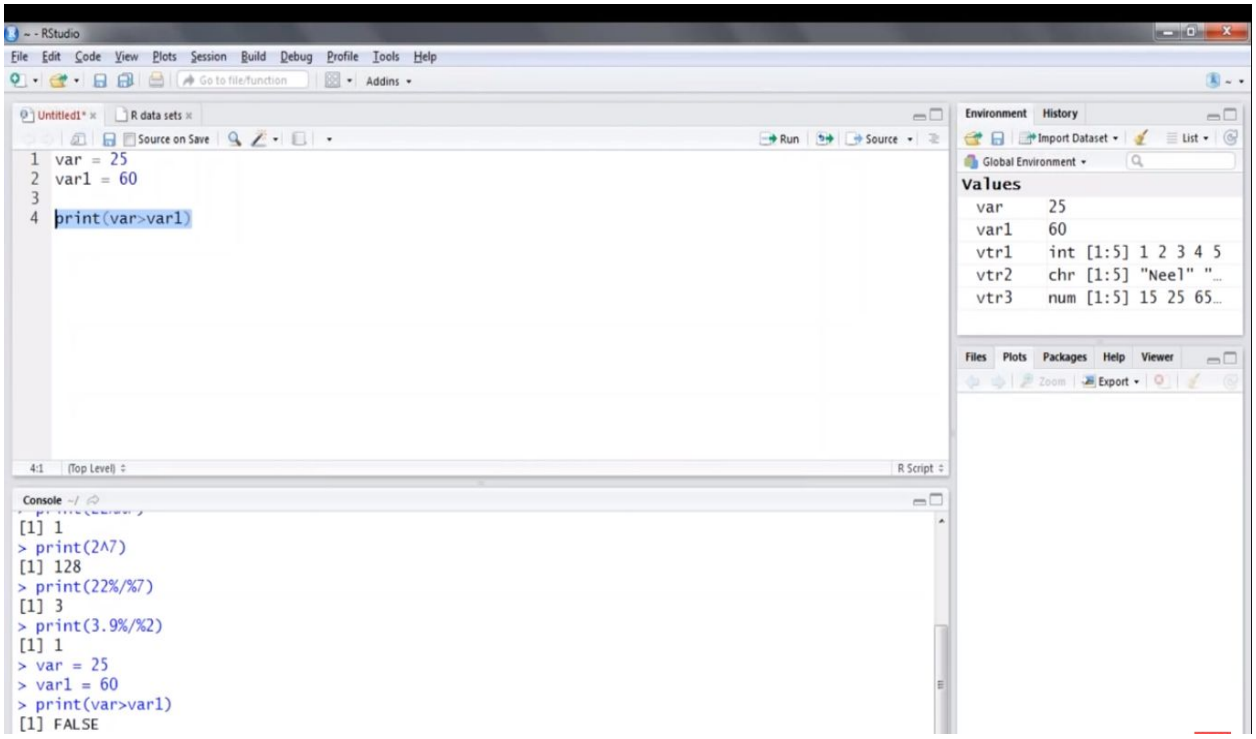


Data Operators In R

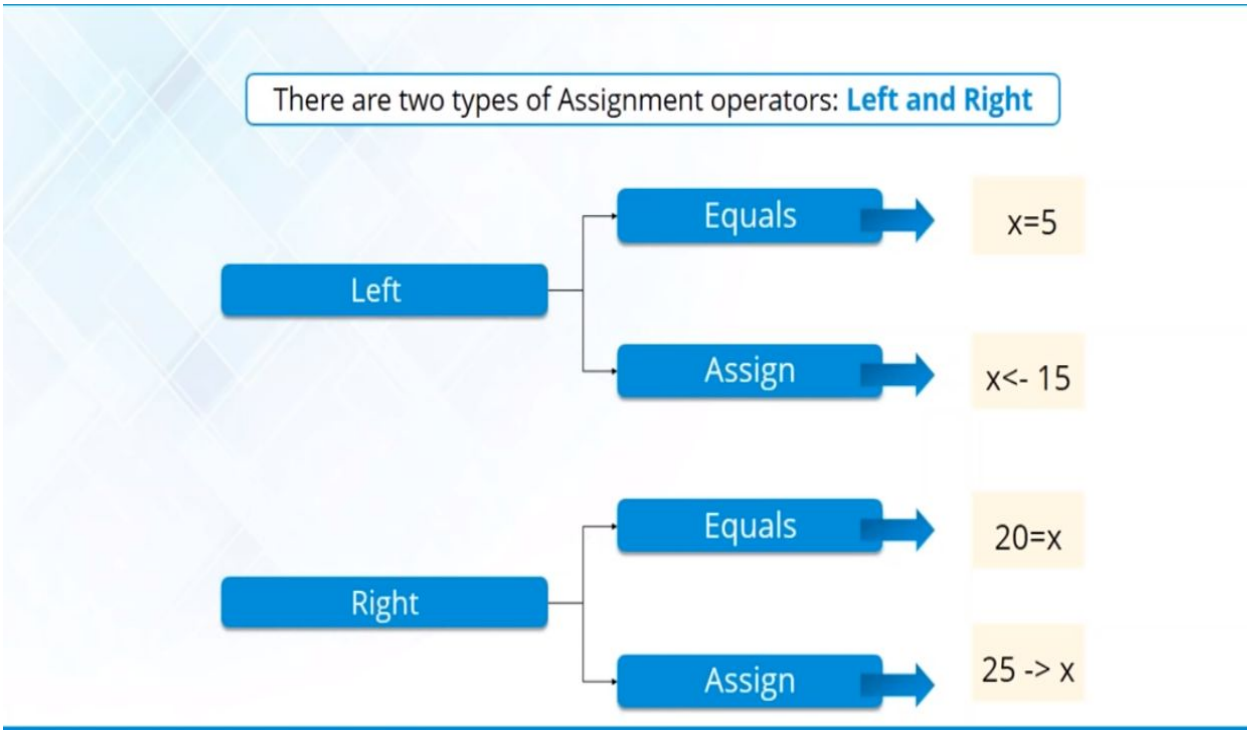


Arithmetic and Relational Operators

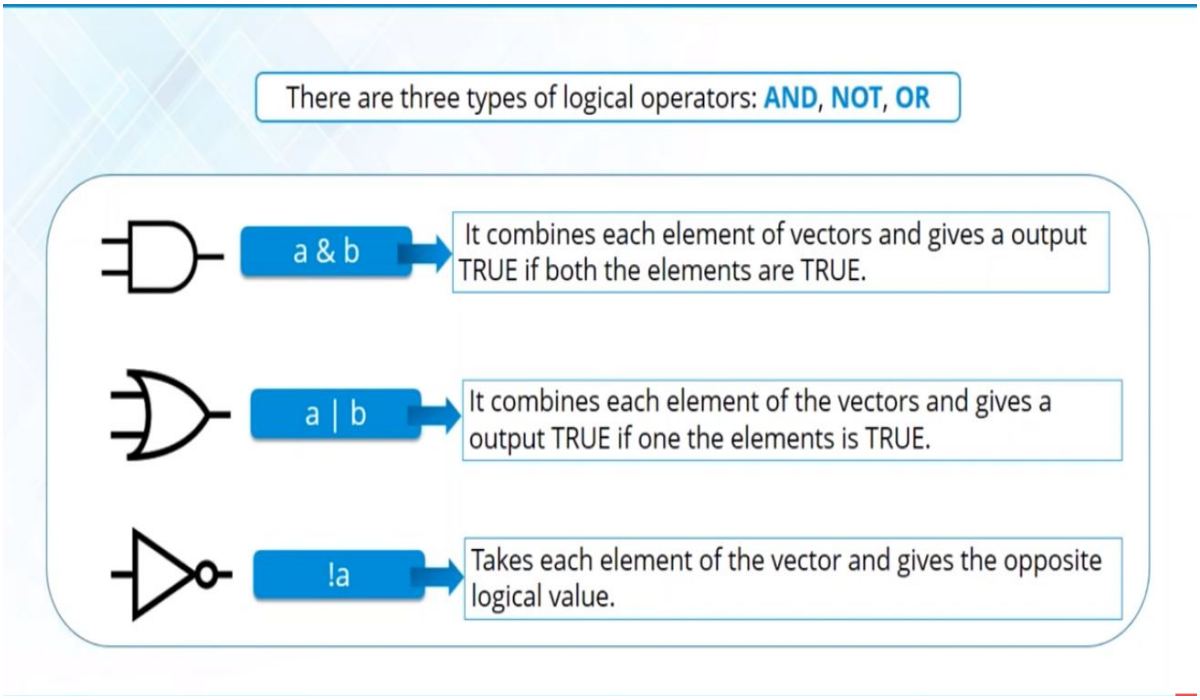


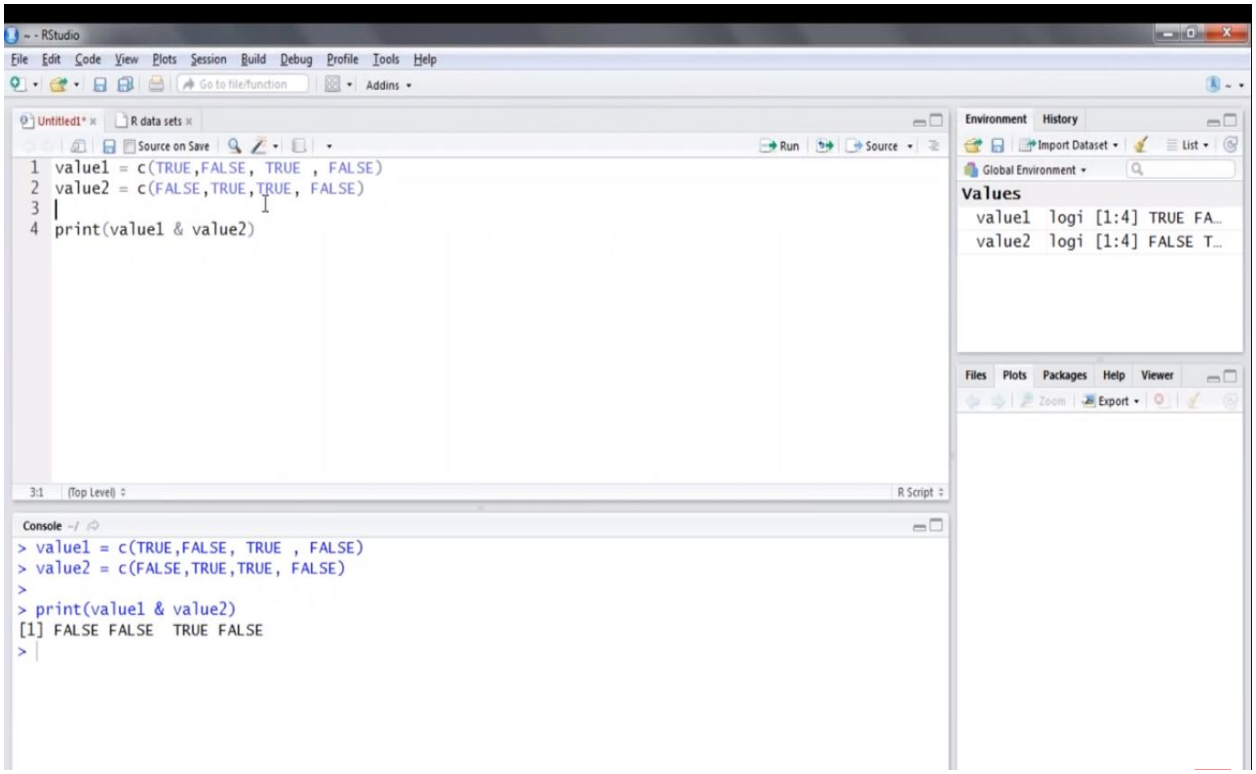


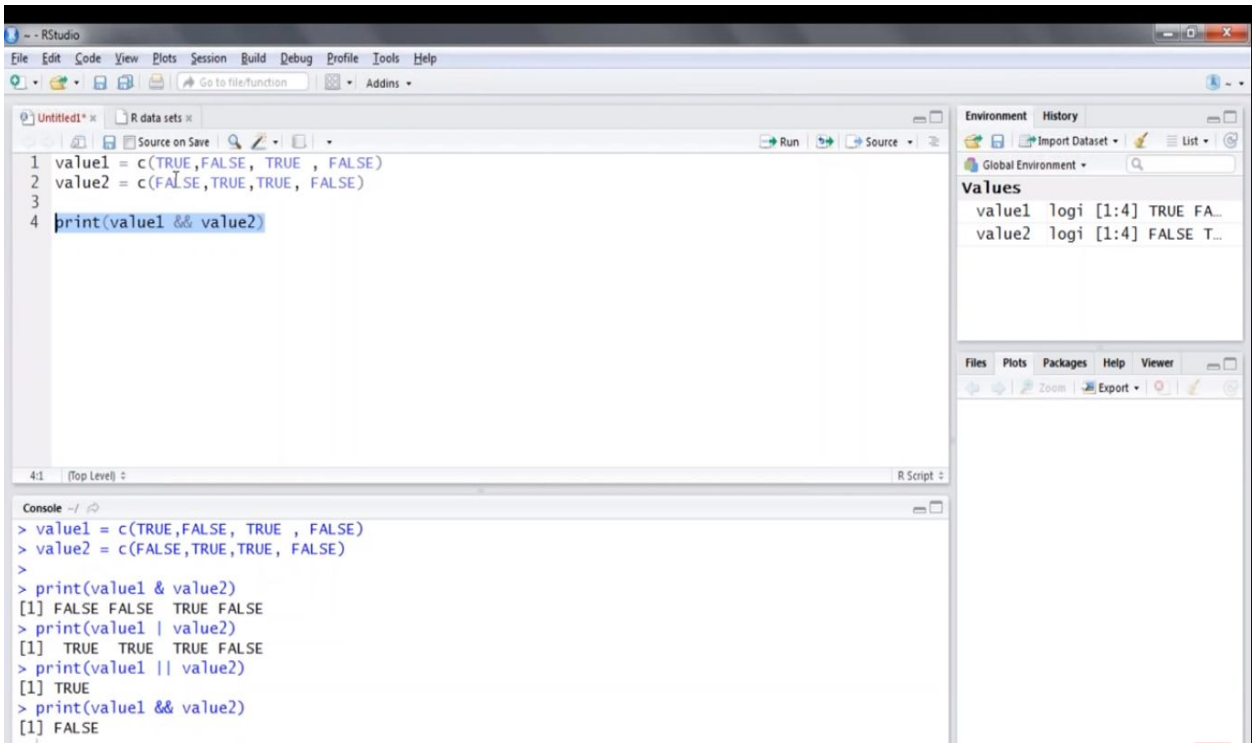
Assignment Operator

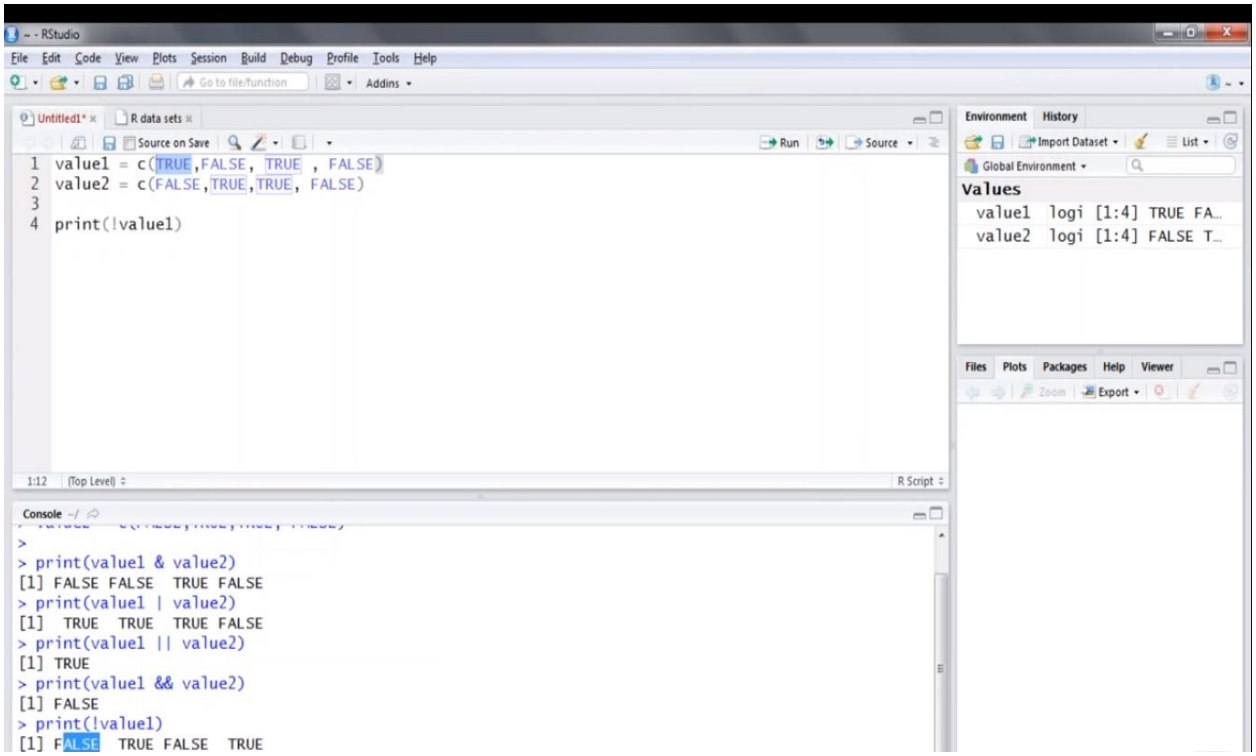


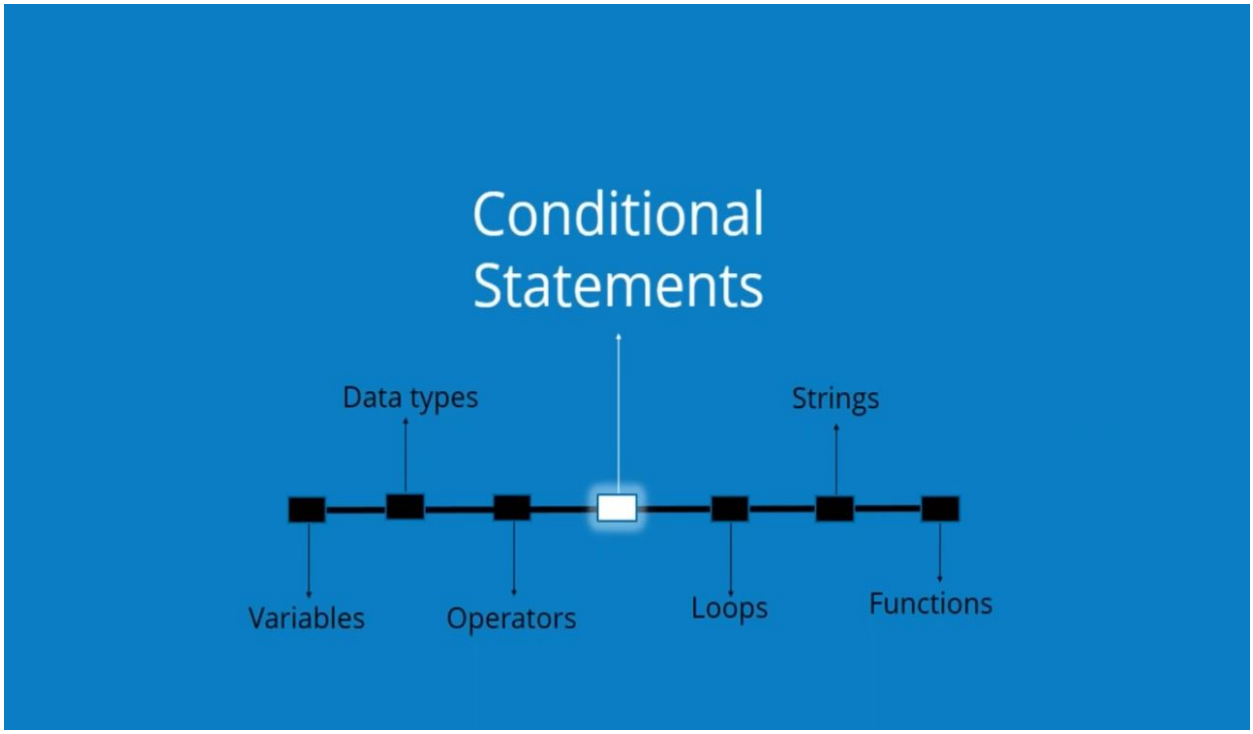
Logical Operator



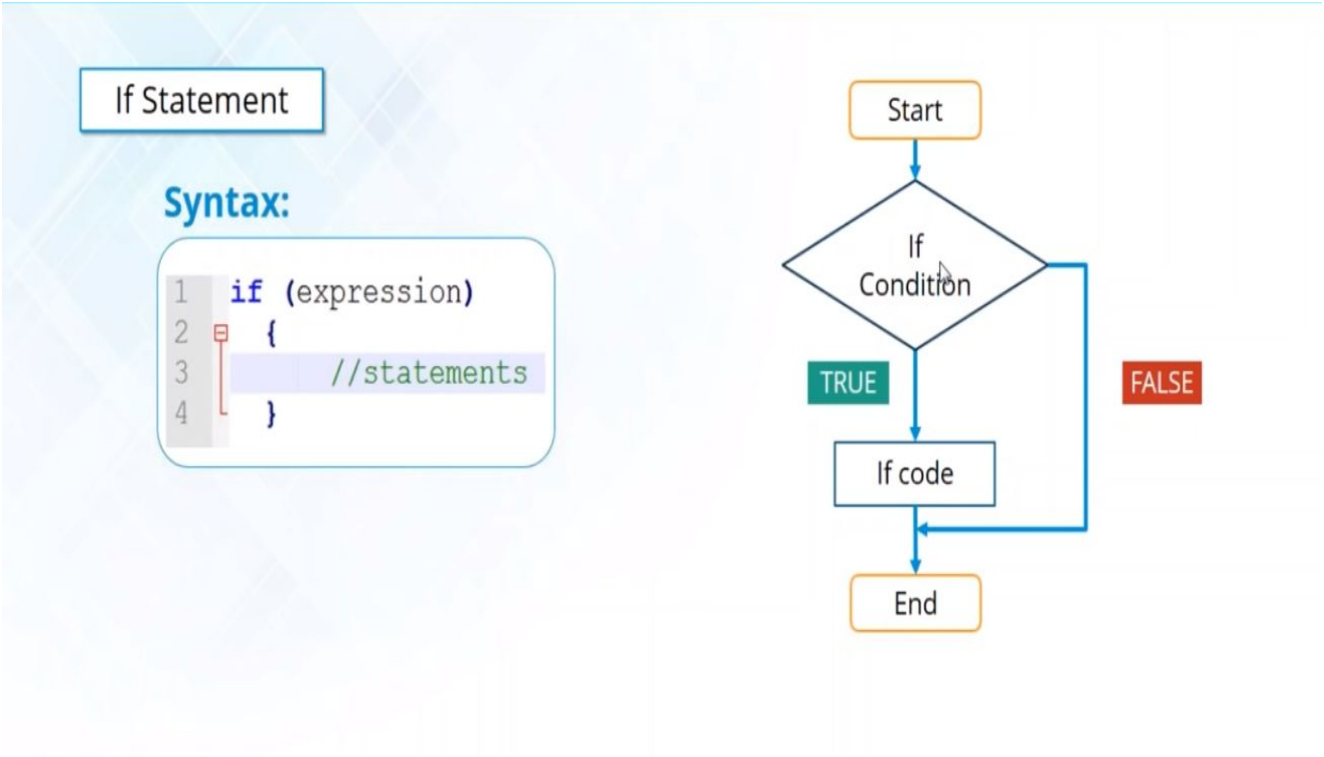

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
R data sets
Source on Save
Run
Source
1 value1 = c(TRUE,FALSE, TRUE , FALSE)
2 value2 = c(FALSE,TRUE,TRUE, FALSE)
3
4 print(value1 & value2)
3:1 (Top Level)
R Script
Console ~/
> value1 = c(TRUE,FALSE, TRUE , FALSE)
> value2 = c(FALSE,TRUE,TRUE, FALSE)
>
> print(value1 & value2)
[1] FALSE FALSE TRUE FALSE
>
Environment History
Import Dataset
List
Global Environment
Values
value1 logi [1:4] TRUE FA…
value2 logi [1:4] FALSE T…
Files Plots Packages Help Viewer
Zoom
Export

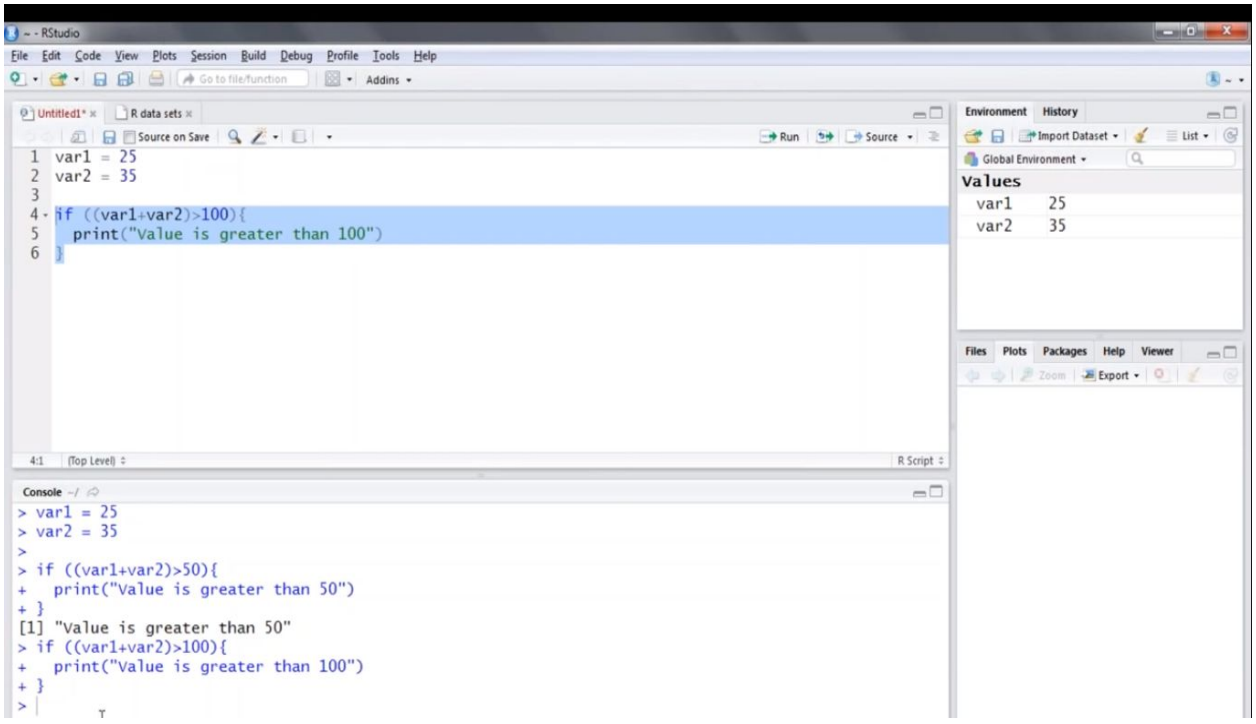

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
R data sets
Source on Save
Run
Source
1 value1 = c(TRUE,FALSE, TRUE , FALSE)
2 value2 = c(FALSE,TRUE,TRUE, FALSE)
3
4 print(value1 && value2)
4:1 (Top Level)
R Script
Console ~/
> value1 = c(TRUE,FALSE, TRUE , FALSE)
> value2 = c(FALSE,TRUE,TRUE, FALSE)
>
> print(value1 & value2)
[1] FALSE FALSE TRUE FALSE
> print(value1 | value2)
[1] TRUE TRUE TRUE FALSE
> print(value1 || value2)
[1] TRUE
> print(value1 && value2)
[1] FALSE
Environment History
Import Dataset
List
Global Environment
Values
value1 logi [1:4] TRUE FA…
value2 logi [1:4] FALSE T…
Files Plots Packages Help Viewer
Zoom
Export





Conditional Statements



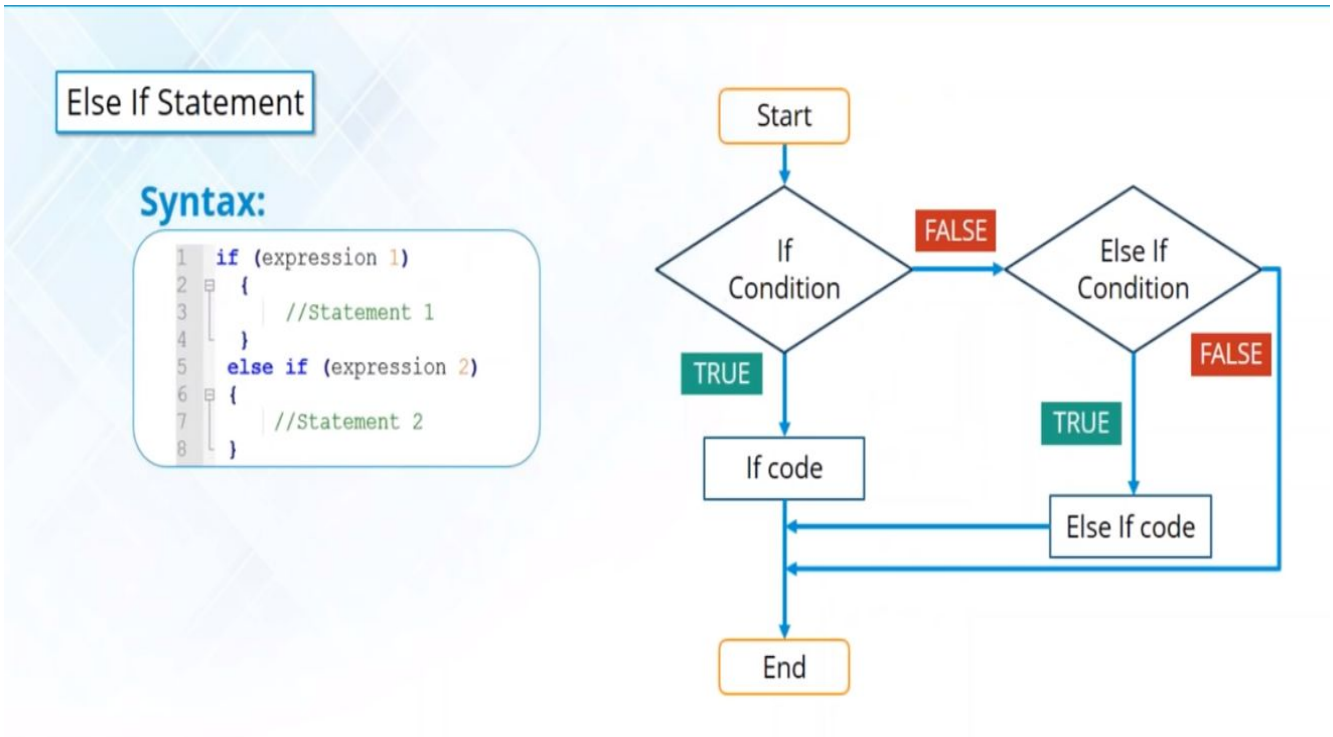


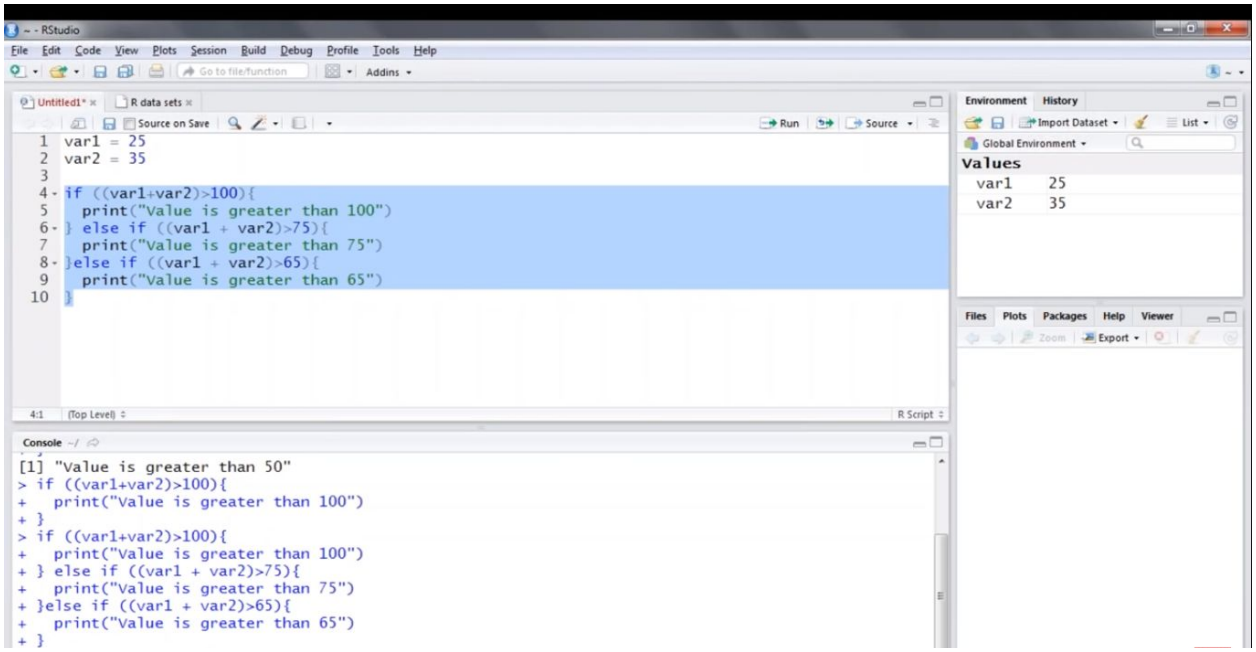
Conditional Statements

if...else statement

The syntax of if...else statement is:

```
if (test_expression) {
statement1
} else {
statement2
}
```





Conditional Statements

if...else Ladder

The if...else ladder (if...else...if) statement allows you execute a block of code among more than 2 alternatives

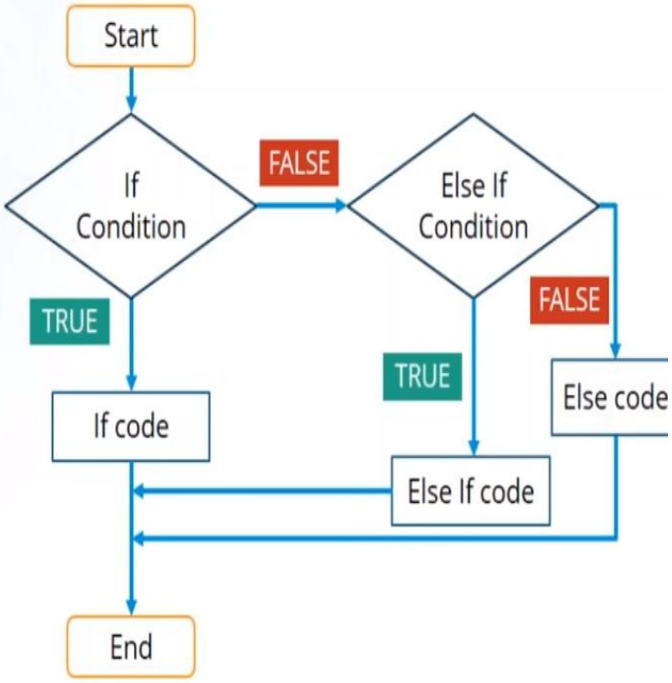
The syntax of if...else statement is:

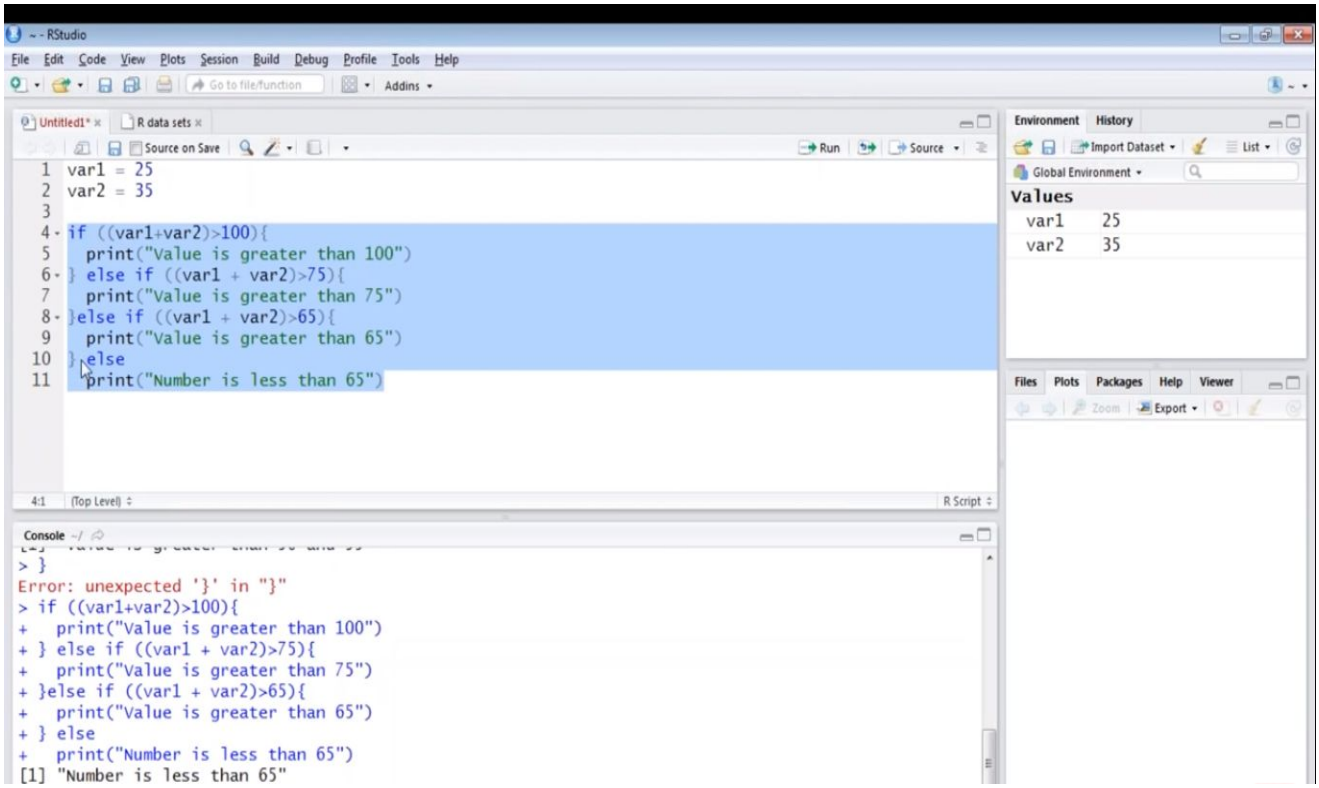
```
if ( test_expression1) {
statement1
} else if ( test_expression2) {
statement2
} else if ( test_expression3) {
statement3
} else {
statement4
}
```

Else Statement

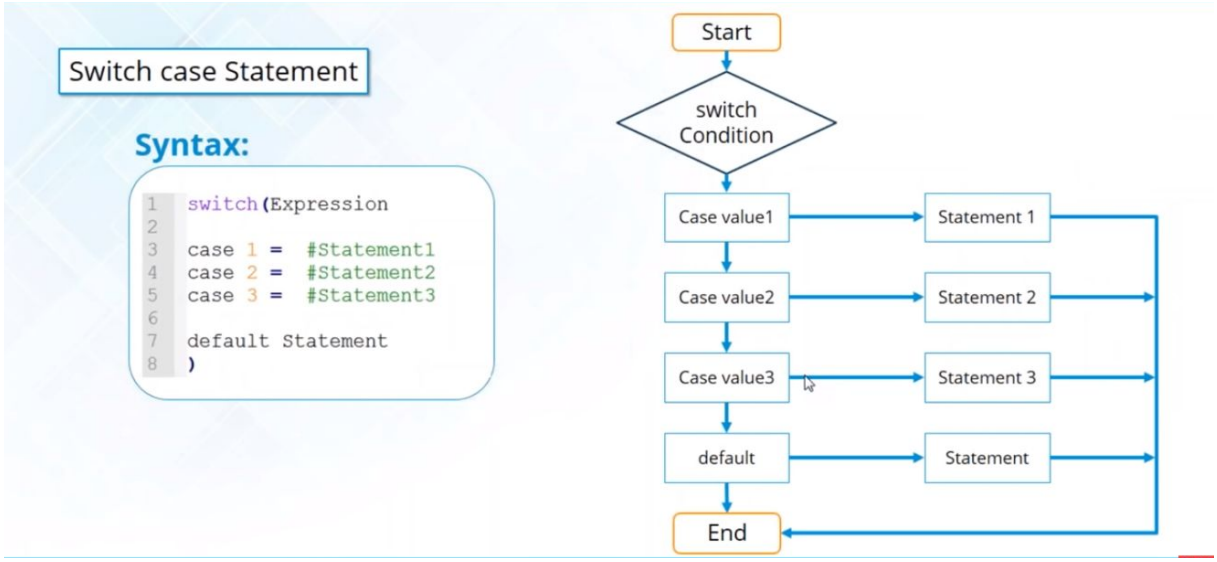
Syntax:

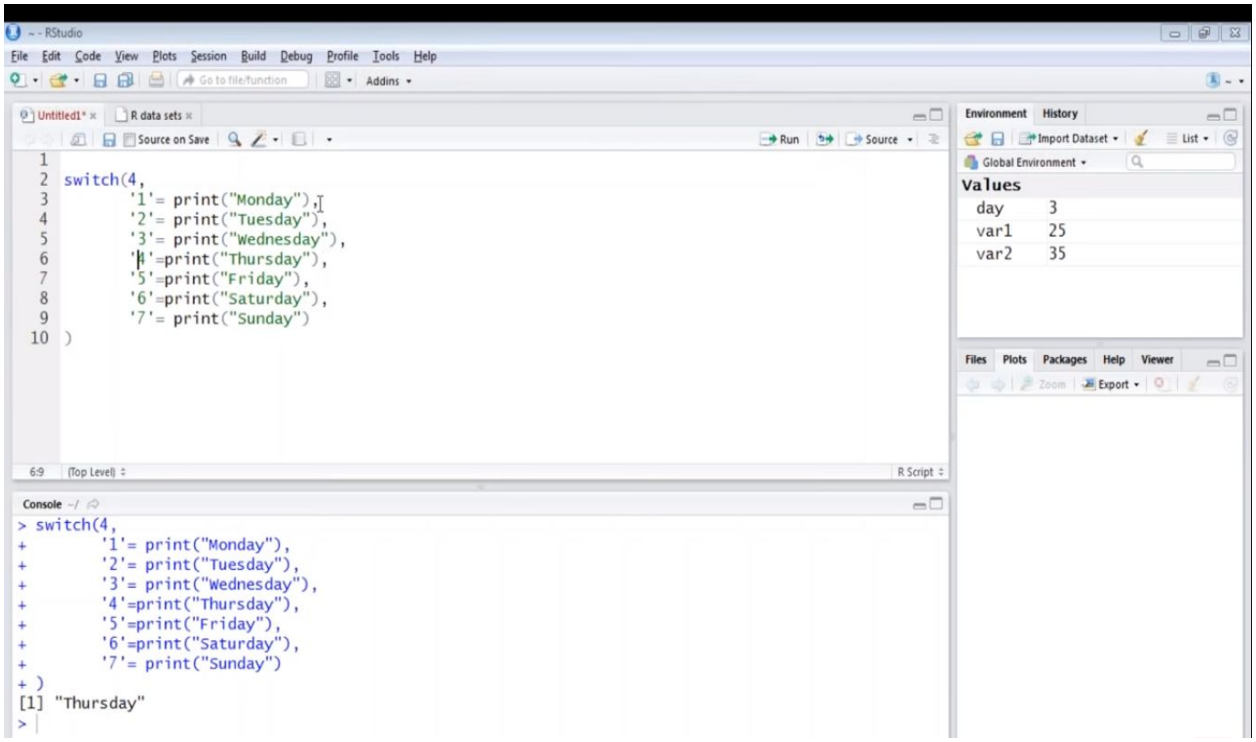
```
if (expression 1)
 {
    //Statement 1
 }
else if (expression 2)
{
    //Statement 2
}
else
{
    //Statement 3
}
```

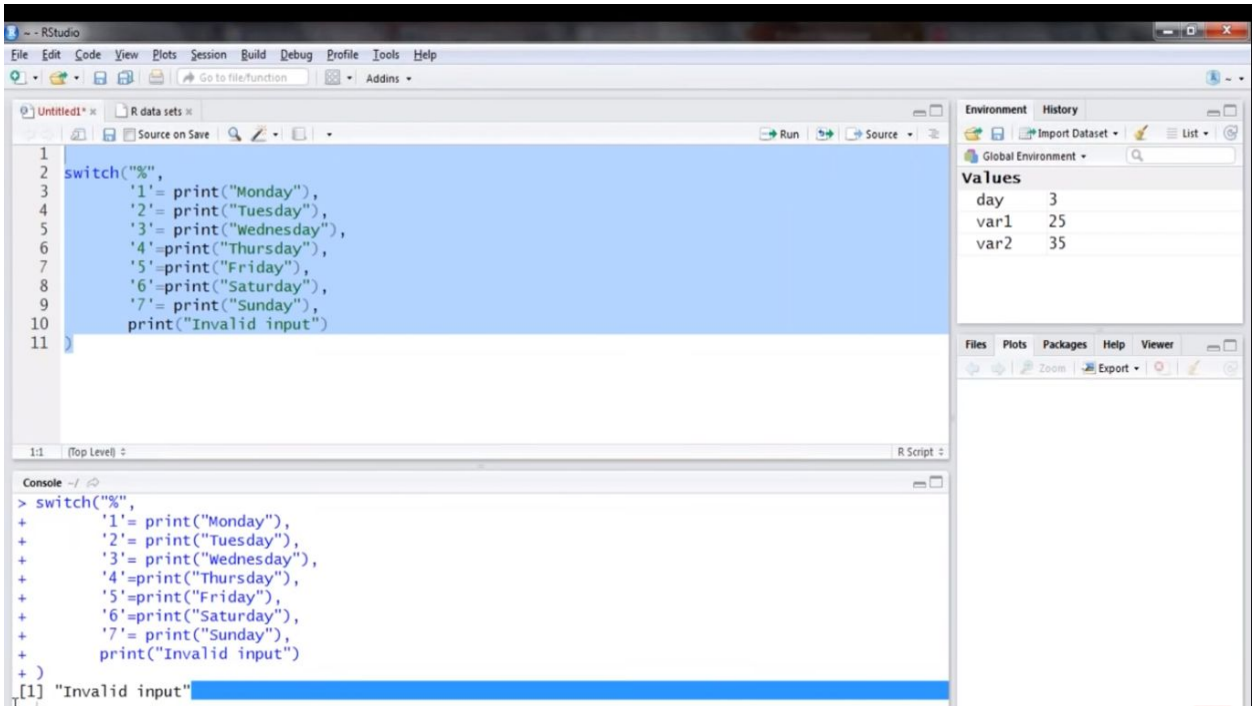




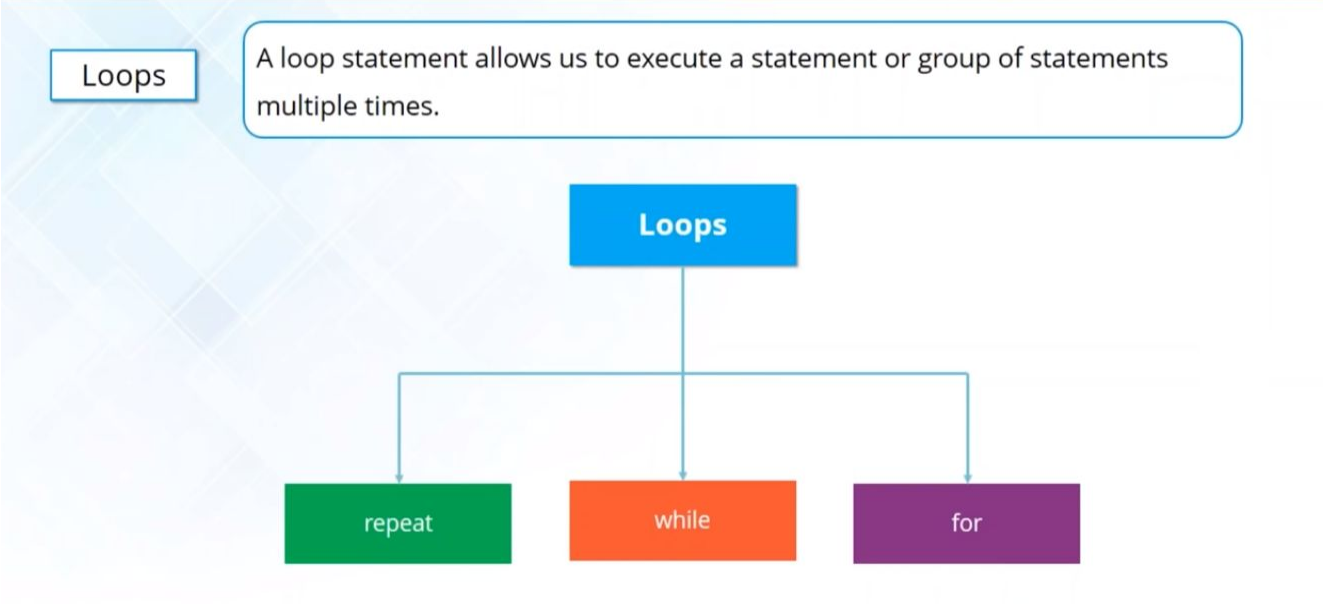
Switch Statement



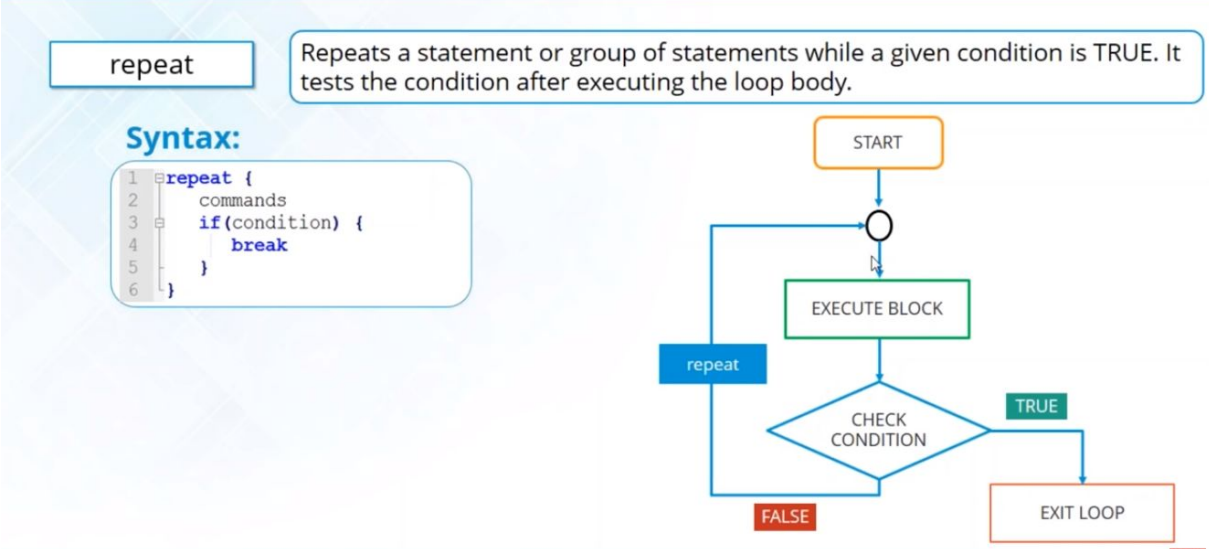

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1* R data sets
Source on Save Run Source
switch(4,
'1'= print("Monday"),
'2'= print("Tuesday"),
'3'= print("Wednesday"),
'4'=print("Thursday"),
'5'=print("Friday"),
'6'=print("Saturday"),
'7'= print("Sunday")
)
6:9 (Top Level) R Script
Console ~/
> switch(4,
+ '1'= print("Monday"),
+ '2'= print("Tuesday"),
+ '3'= print("Wednesday"),
+ '4'=print("Thursday"),
+ '5'=print("Friday"),
+ '6'=print("Saturday"),
+ '7'= print("Sunday")
+)
[1] "Thursday"
Environment History
Import Dataset List
Global Environment
Values
day 3
var1 25
var2 35
Files Plots Packages Help Viewer
Zoom Export

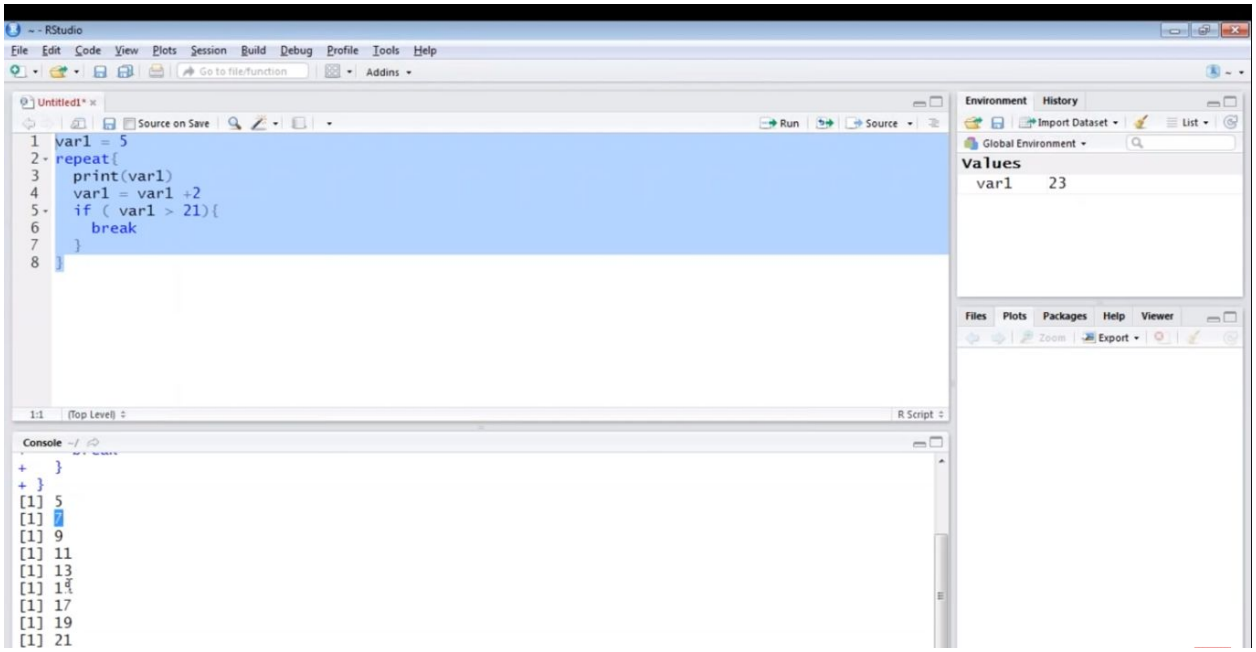

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1* R data sets
Source on Save Run Source
switch("%",
'1'= print("Monday"),
'2'= print("Tuesday"),
'3'= print("Wednesday"),
'4'=print("Thursday"),
'5'=print("Friday"),
'6'=print("Saturday"),
'7'= print("Sunday"),
print("Invalid input")
)
1:1 (Top Level) R Script
Console ~/
> switch("%",
+ '1'= print("Monday"),
+ '2'= print("Tuesday"),
+ '3'= print("Wednesday"),
+ '4'=print("Thursday"),
+ '5'=print("Friday"),
+ '6'=print("Saturday"),
+ '7'= print("Sunday"),
+ print("Invalid input")
+)
[1] "Invalid input"
Environment History
Import Dataset List
Global Environment
Values
day 3
var1 25
var2 35
Files Plots Packages Help Viewer
Zoom Export

Loops In R

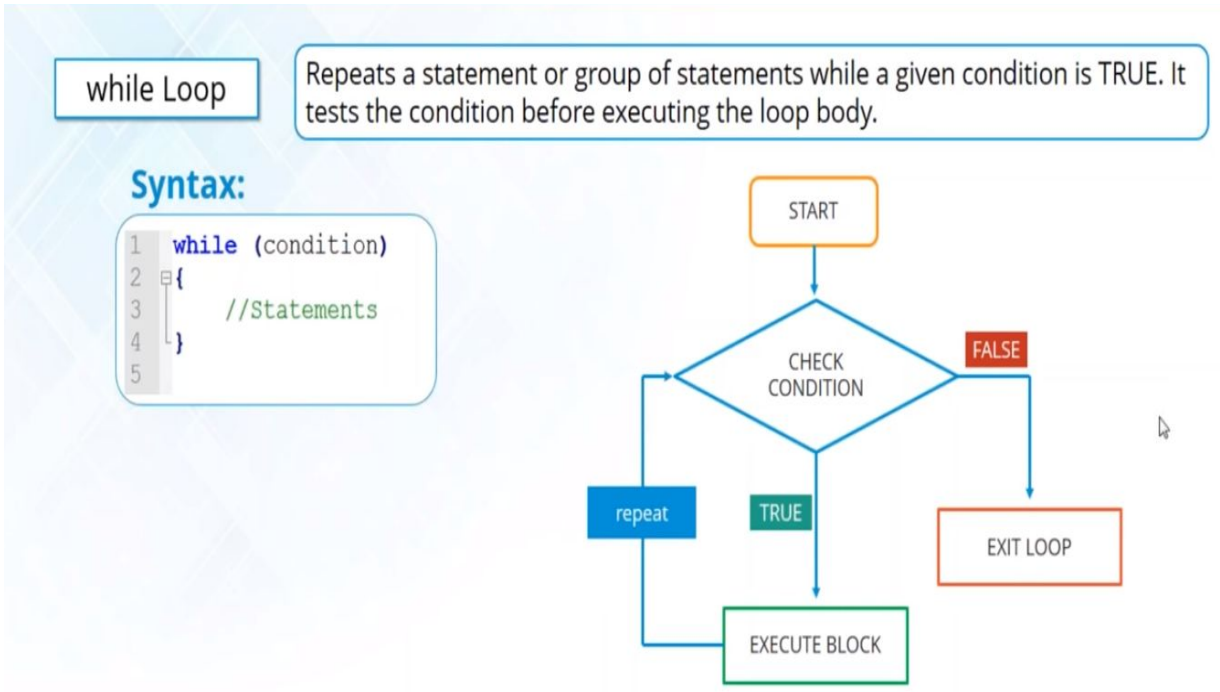


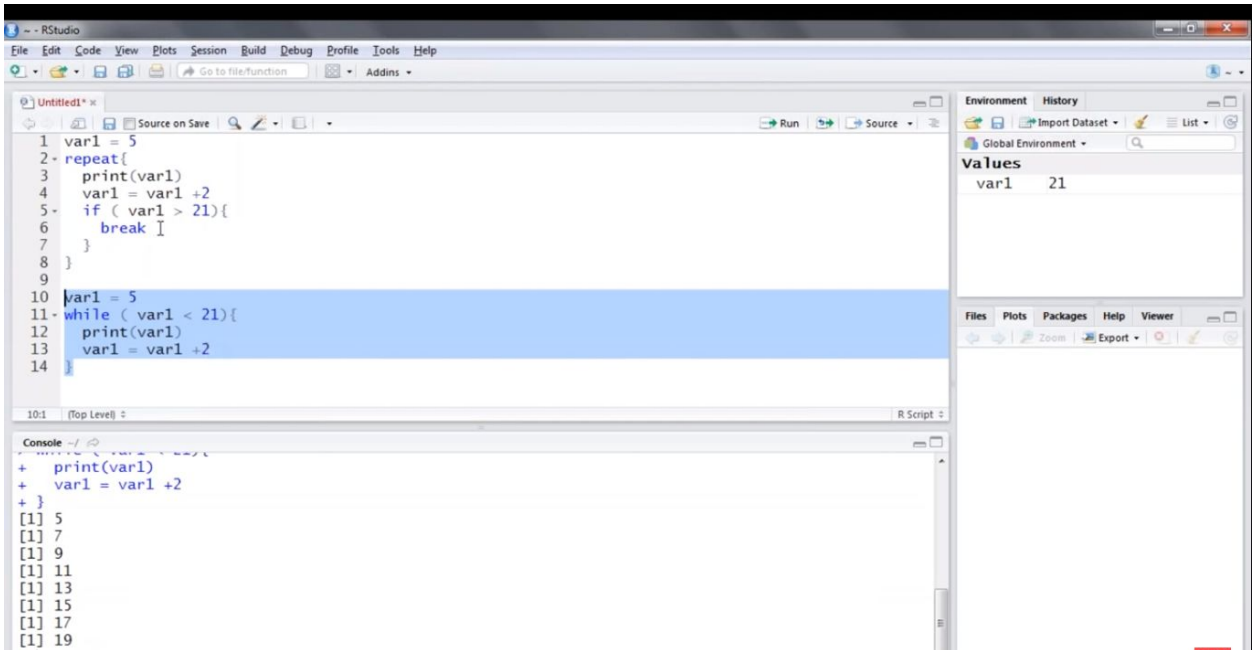
Repeat Loop

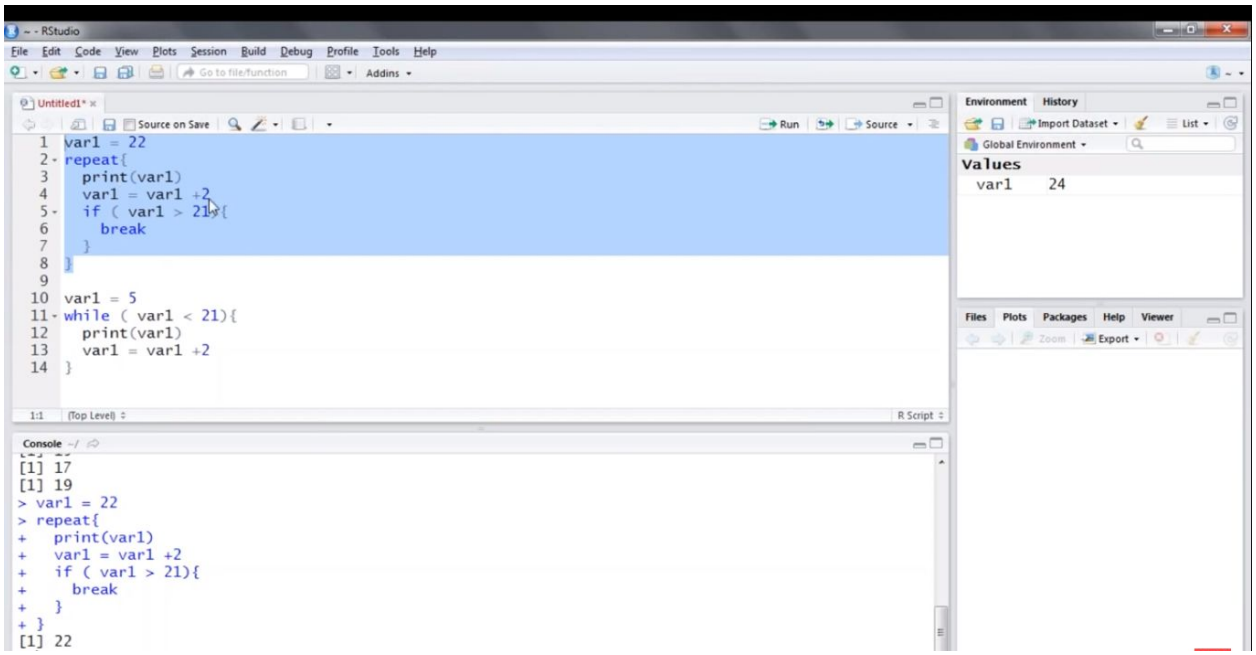



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
var1 = 5
repeat{
print(var1)
var1 = var1 +2
if (var1 > 21){
break
}
}
Environment History
Import Dataset
Global Environment
Values
var1 23
Files Plots Packages Help Viewer
Export
Console
+ }
+ }
[1] 5
[1] 7
[1] 9
[1] 11
[1] 13
[1] 15
[1] 17
[1] 19
[1] 21

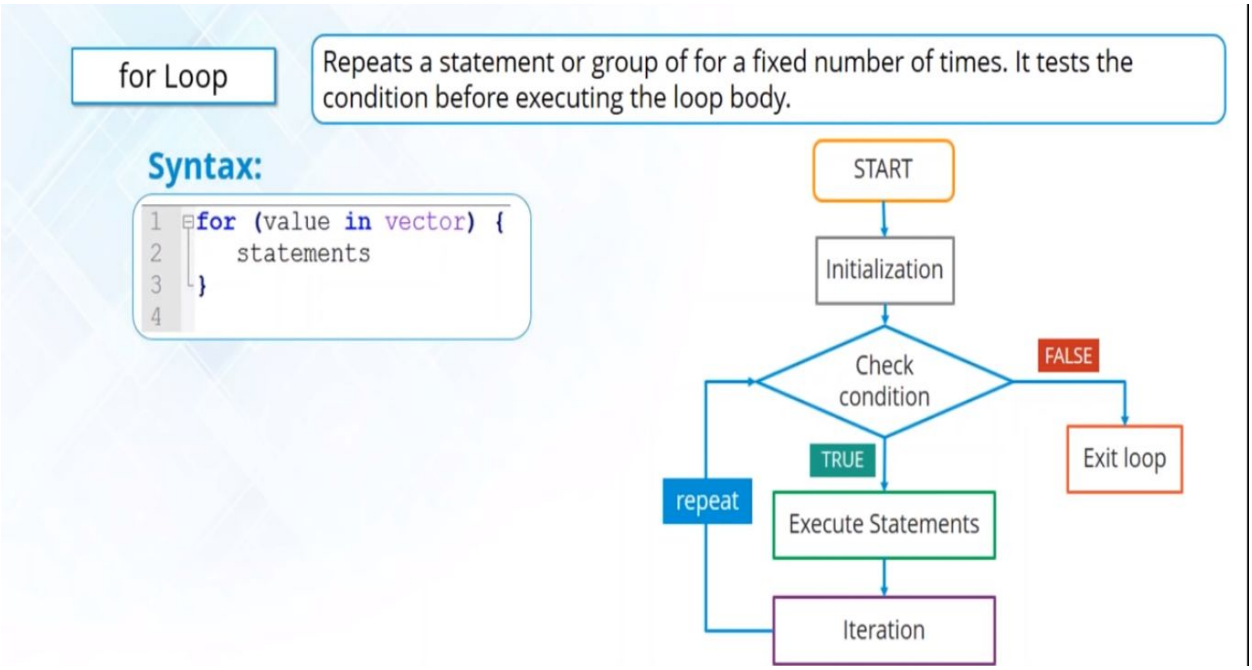
While loop

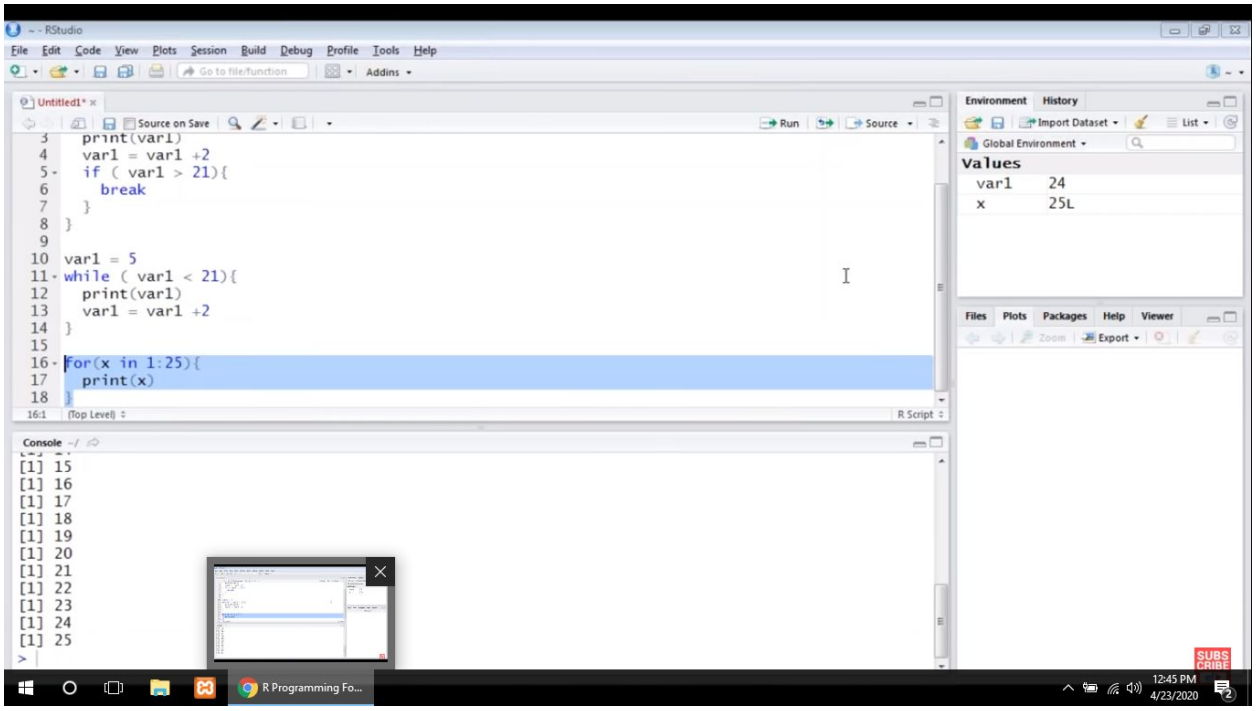

while Loop
Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body.
Syntax:
while (condition)
{
//Statements
}
START
CHECK CONDITION
FALSE
TRUE
repeat
EXIT LOOP
EXECUTE BLOCK


~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Addins
Untitled1*
Source on Save
Run
Source
1 var1 = 5
2 repeat{
3 print(var1)
4 var1 = var1 +2
5 if (var1 > 21){
6 break
7 }
8 }
9
10 var1 = 5
11 while (var1 < 21){
12 print(var1)
13 var1 = var1 +2
14 }
10:1 (Top Level)
R Script
Console ~/
+ print(var1)
+ var1 = var1 +2
+ }
[1] 5
[1] 7
[1] 9
[1] 11
[1] 13
[1] 15
[1] 17
[1] 19
Environment History
Import Dataset
List
Global Environment
Values
var1 21
Files Plots Packages Help Viewer
Zoom
Export


~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Addins
Untitled1*
Source on Save
Run
Source
1 var1 = 22
2 repeat{
3 print(var1)
4 var1 = var1 +2
5 if (var1 > 21){
6 break
7 }
8 }
9
10 var1 = 5
11 while (var1 < 21){
12 print(var1)
13 var1 = var1 +2
14 }
1:1 (Top Level)
R Script
Console ~/
[1] 17
[1] 19
> var1 = 22
> repeat{
+ print(var1)
+ var1 = var1 +2
+ if (var1 > 21){
+ break
+ }
+ }
[1] 22
Environment History
Import Dataset
List
Global Environment
Values
var1 24
Files Plots Packages Help Viewer
Zoom
Export

For Loop


for Loop
Repeats a statement or group of for a fixed number of times. It tests the condition before executing the loop body.
Syntax:
for (value in vector) {
statements
}
START
Initialization
Check condition
FALSE
Exit loop
TRUE
repeat
Execute Statements
Iteration


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
print(var1)
var1 = var1 +2
if (var1 > 21){
break
}
}
var1 = 5
while (var1 < 21){
print(var1)
var1 = var1 +2
}
for(x in 1:25){
print(x)
}
Environment History
Import Dataset
Global Environment
Values
var1 24
x 25L
Files Plots Packages Help Viewer
Export
Console
[1] 15
[1] 16
[1] 17
[1] 18
[1] 19
[1] 20
[1] 21
[1] 22
[1] 23
[1] 24
[1] 25
R Programming Fo...
12:45 PM
4/23/2020

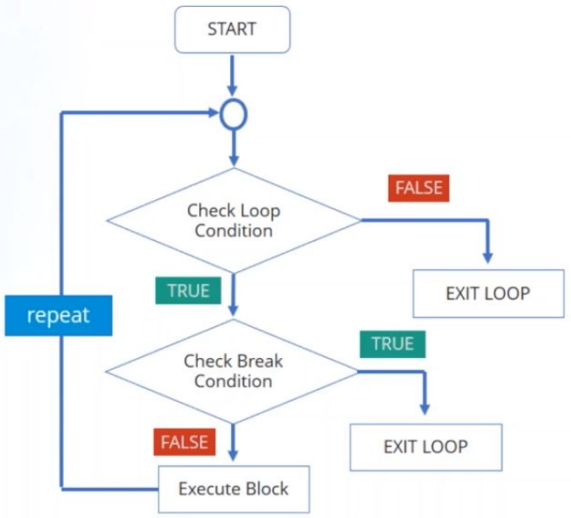
Flow Control

edureka!

7. break

Syntax:

```
break;
```



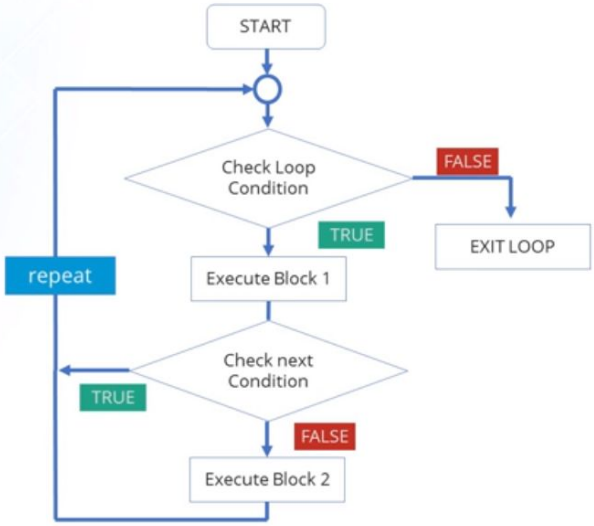
Flow Control

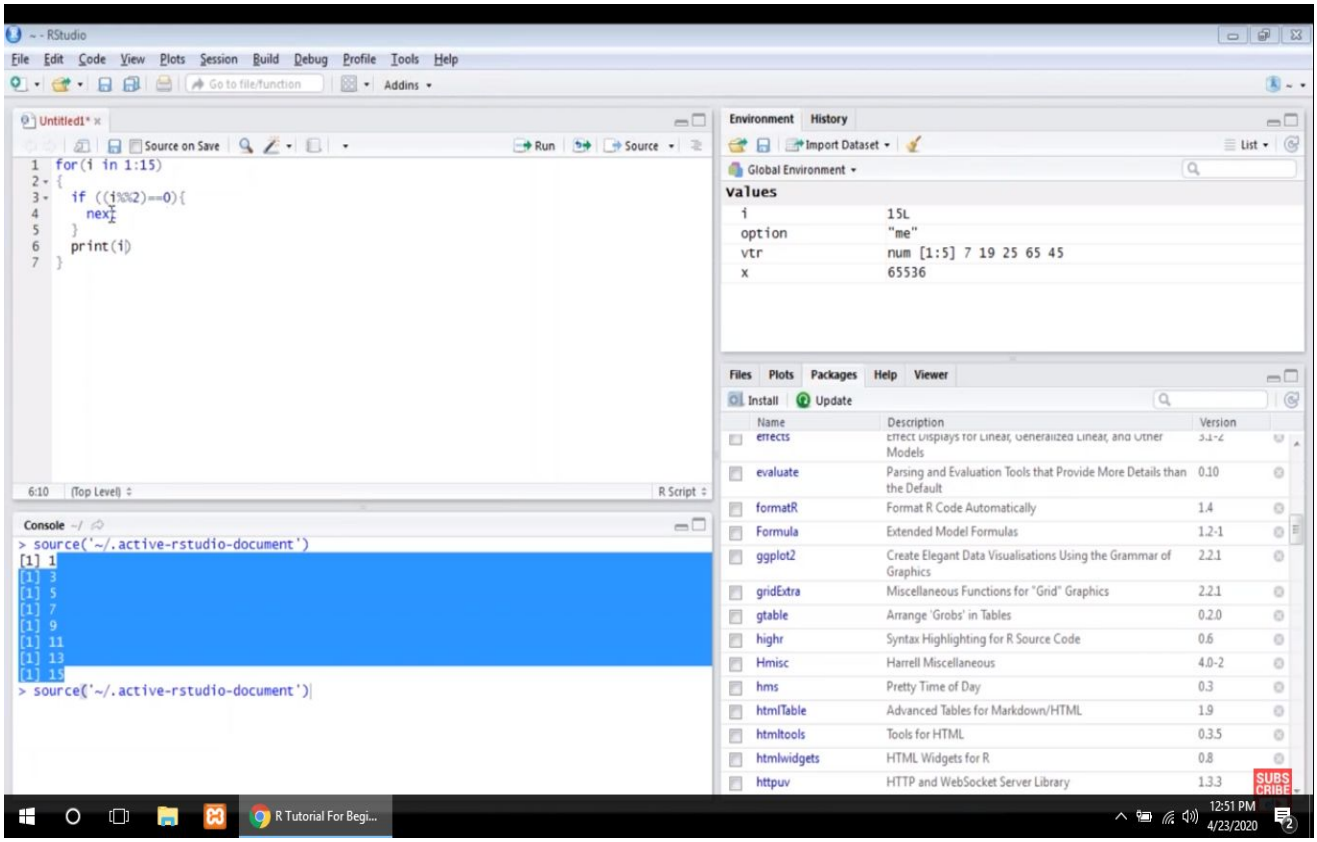
edureka!

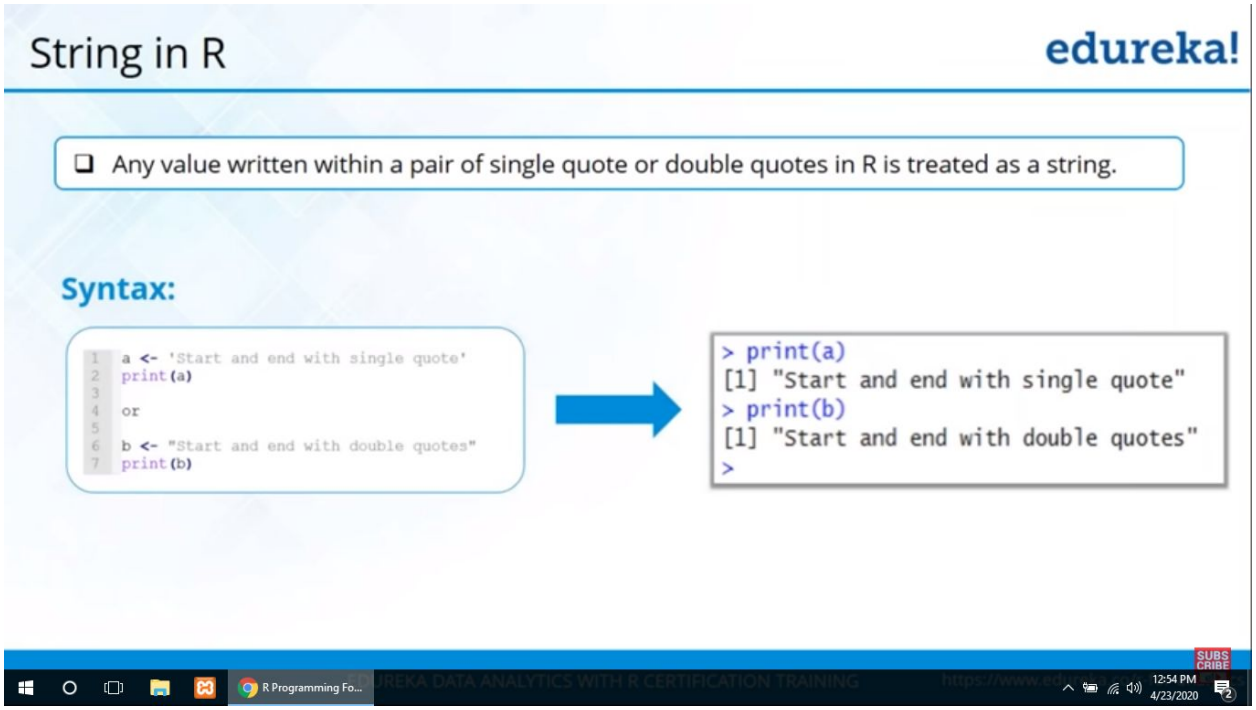
8. next

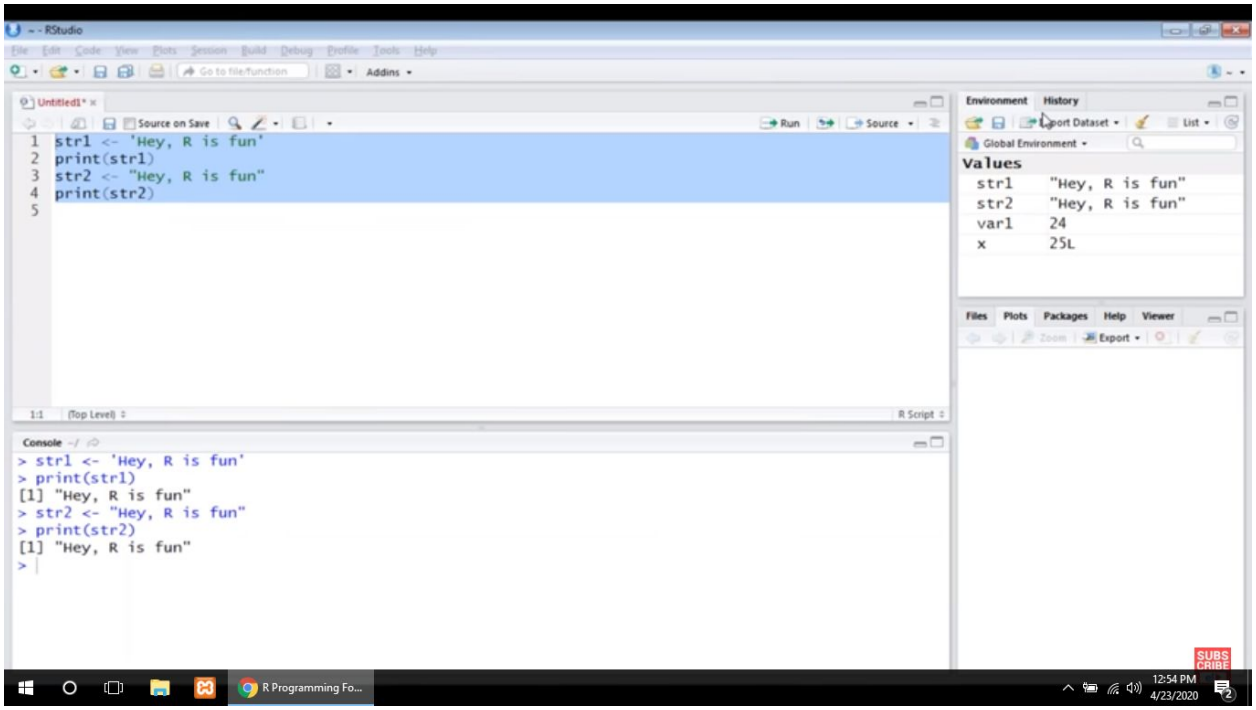
Syntax:

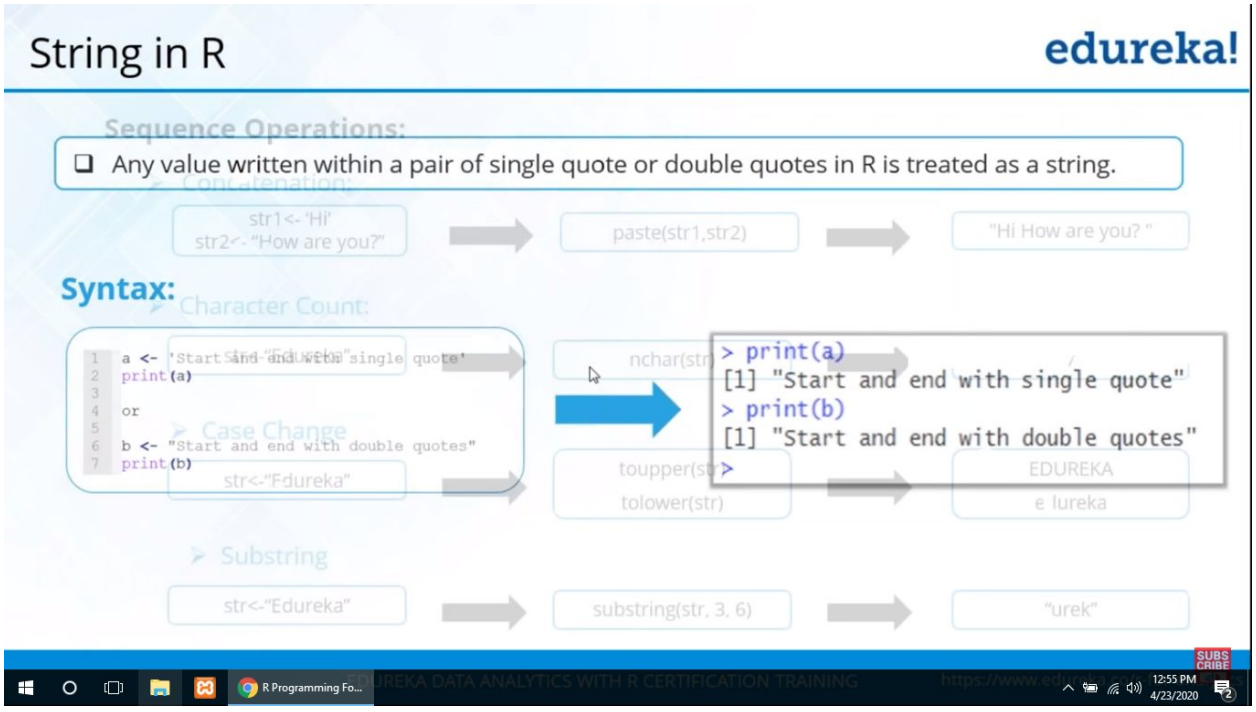
```
next;
```

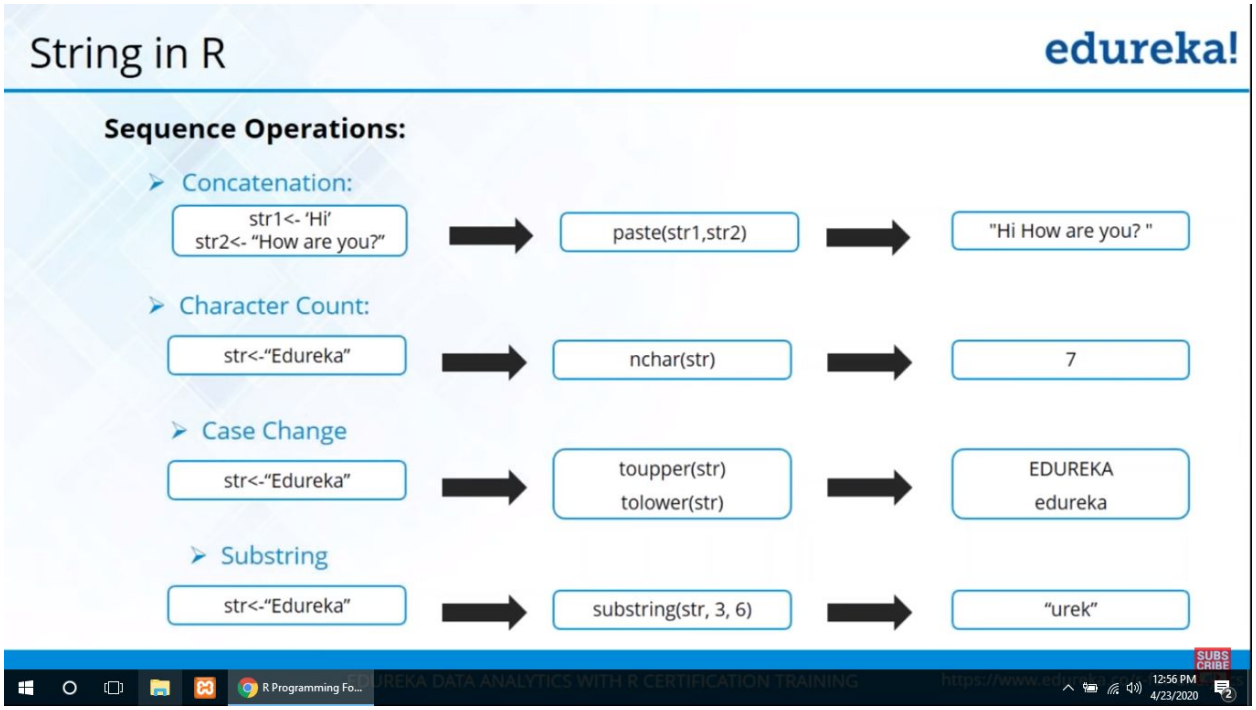


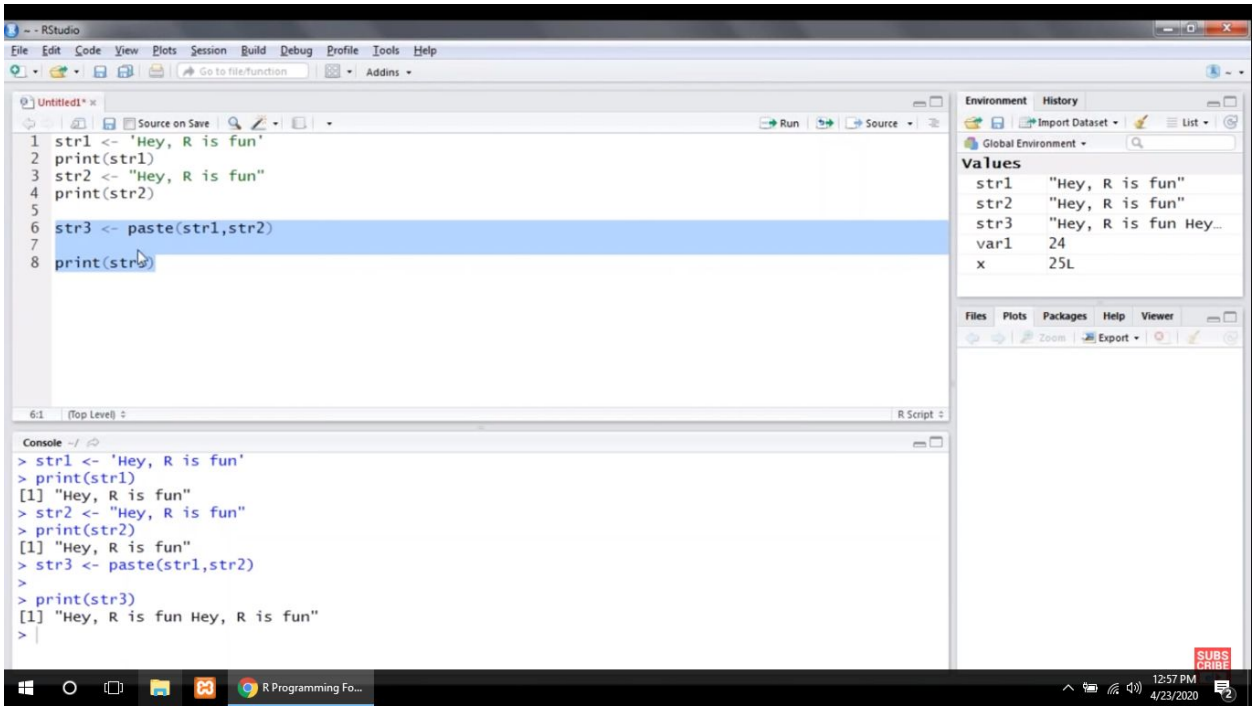

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
for(i in 1:15)
{
if ((i%%2)==0){
next
}
print(i)
}
6:10 (Top Level)
R Script
Environment History
Import Dataset
List
Global Environment
values
i 15L
option "me"
vtr num [1:5] 7 19 25 65 45
x 65536
Files Plots Packages Help Viewer
Install Update
Name Description Version
effects Effect Displays for Linear, Generalized Linear, and Other Models
evaluate Parsing and Evaluation Tools that Provide More Details than the Default 0.10
formatR Format R Code Automatically 1.4
Formula Extended Model Formulas 1.2-1
ggplot2 Create Elegant Data Visualisations Using the Grammar of Graphics 2.2.1
gridExtra Miscellaneous Functions for "Grid" Graphics 2.2.1
gtable Arrange 'Grobs' in Tables 0.2.0
highr Syntax Highlighting for R Source Code 0.6
Hmisc Harrell Miscellaneous 4.0-2
hms Pretty Time of Day 0.3
htmlTable Advanced Tables for Markdown/HTML 1.9
htmltools Tools for HTML 0.3.5
htmlwidgets HTML Widgets for R 0.8
httpuv HTTP and WebSocket Server Library 1.3.3
Console ~/
> source('~/.active-rstudio-document')
[1] 1
[1] 3
[1] 5
[1] 7
[1] 9
[1] 11
[1] 13
[1] 15
> source('~/.active-rstudio-document')
R Tutorial For Begi...
12:51 PM
4/23/2020

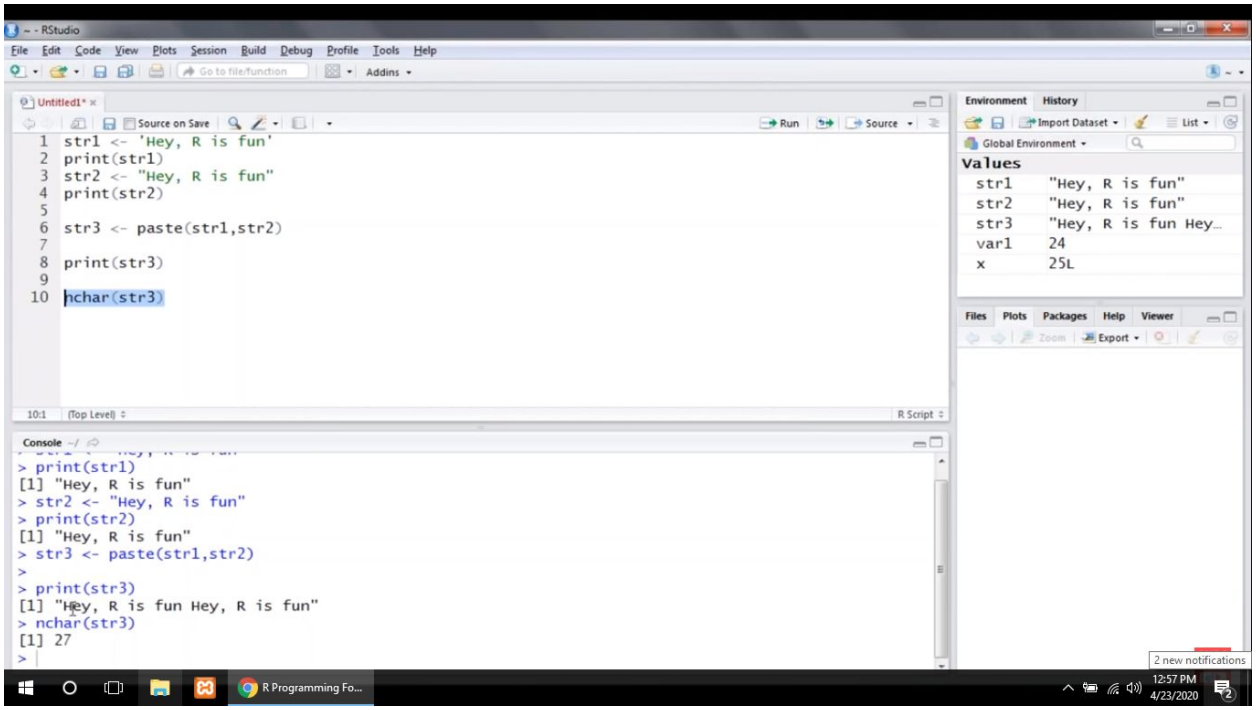

String in R
edureka!
Any value written within a pair of single quote or double quotes in R is treated as a string.
Syntax:
a <- 'Start and end with single quote'
print(a)
or
b <- "Start and end with double quotes"
print(b)
> print(a)
[1] "Start and end with single quote"
> print(b)
[1] "Start and end with double quotes"
>
R Programming Fo...
12:54 PM
4/23/2020

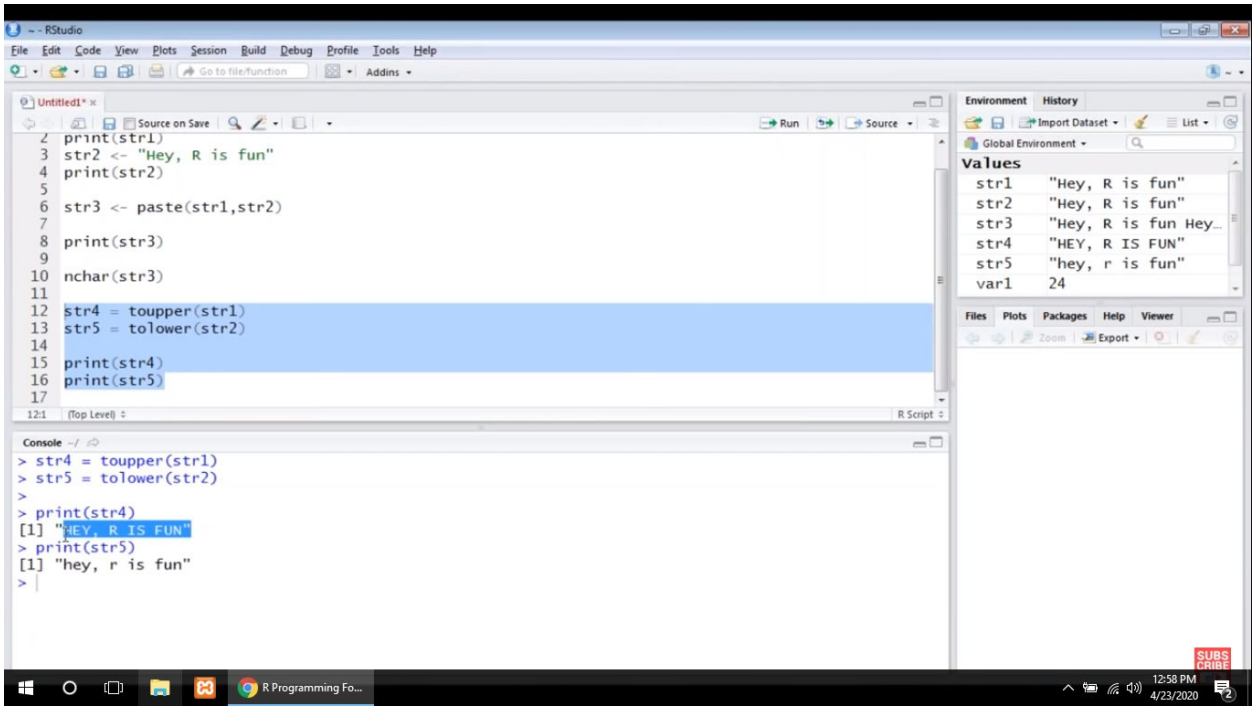

RStudio
Untitled1*
Source on Save
Run
Source
1 str1 <- 'Hey, R is fun'
2 print(str1)
3 str2 <- "Hey, R is fun"
4 print(str2)
5
Environment
History
Import Dataset
List
Global Environment
Values
str1 "Hey, R is fun"
str2 "Hey, R is fun"
var1 24
x 25L
Files
Plots
Packages
Help
Viewer
Export
Console
> str1 <- 'Hey, R is fun'
> print(str1)
[1] "Hey, R is fun"
> str2 <- "Hey, R is fun"
> print(str2)
[1] "Hey, R is fun"
R Programming Fo...
12:54 PM
4/23/2020

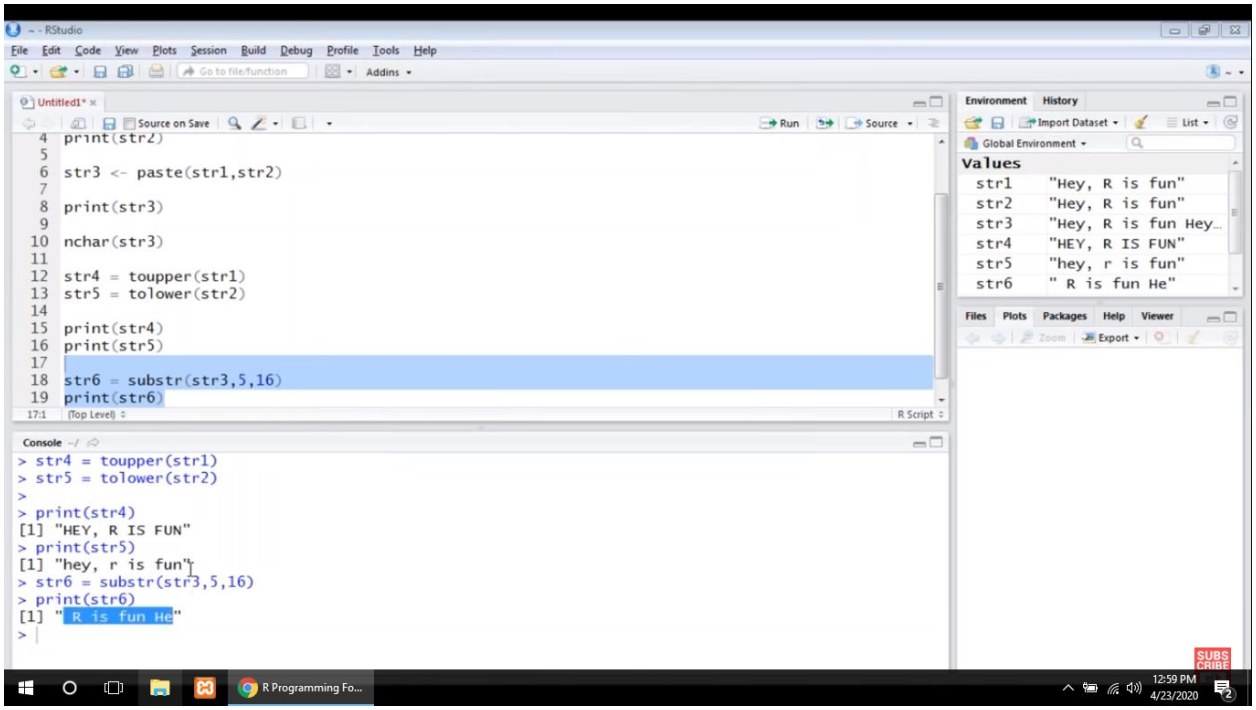

String in R
edureka!
Sequence Operations:
Any value written within a pair of single quote or double quotes in R is treated as a string.
Concatenation:
str1<-'Hi'
str2<-"How are you?"
paste(str1,str2)
"Hi How are you? "
Syntax:
Character Count:
a <- 'Start and end with single quote'
print(a)
or
b <- "Start and end with double quotes"
print(b)
nchar(str
> print(a)
[1] "Start and end with single quote"
> print(b)
[1] "Start and end with double quotes"
>
Case Change
str<-"Fdureka"
toupper(str)
tolower(str)
EDUREKA
e lureka
Substring
str<-"Edureka"
substring(str, 3, 6)
"urek"
R Programming Fo...
12:55 PM
4/23/2020

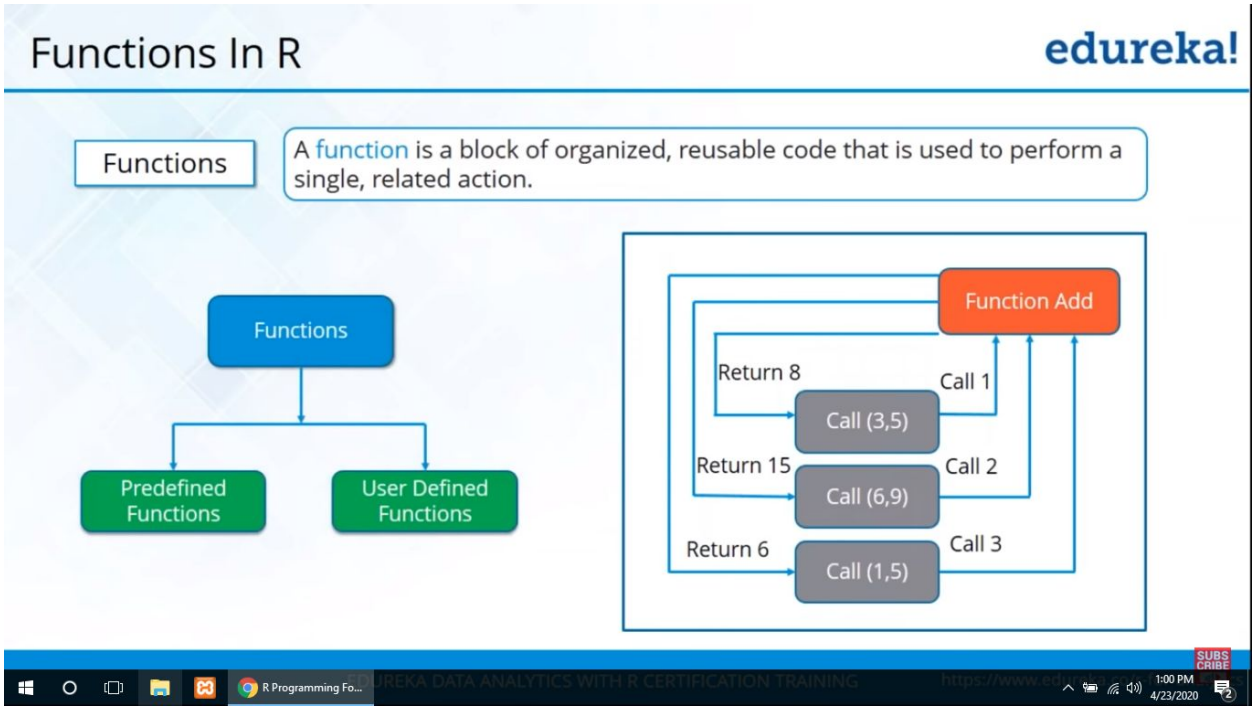

String in R
edureka!
Sequence Operations:
Concatenation:
str1<- 'Hi'
str2<- "How are you?"
paste(str1,str2)
"Hi How are you? "
Character Count:
str<-"Edureka"
nchar(str)
7
Case Change
str<-"Edureka"
toupper(str)
tolower(str)
EDUREKA
edureka
Substring
str<-"Edureka"
substring(str, 3, 6)
"urek"
R Programming Fo...
12:56 PM
4/23/2020

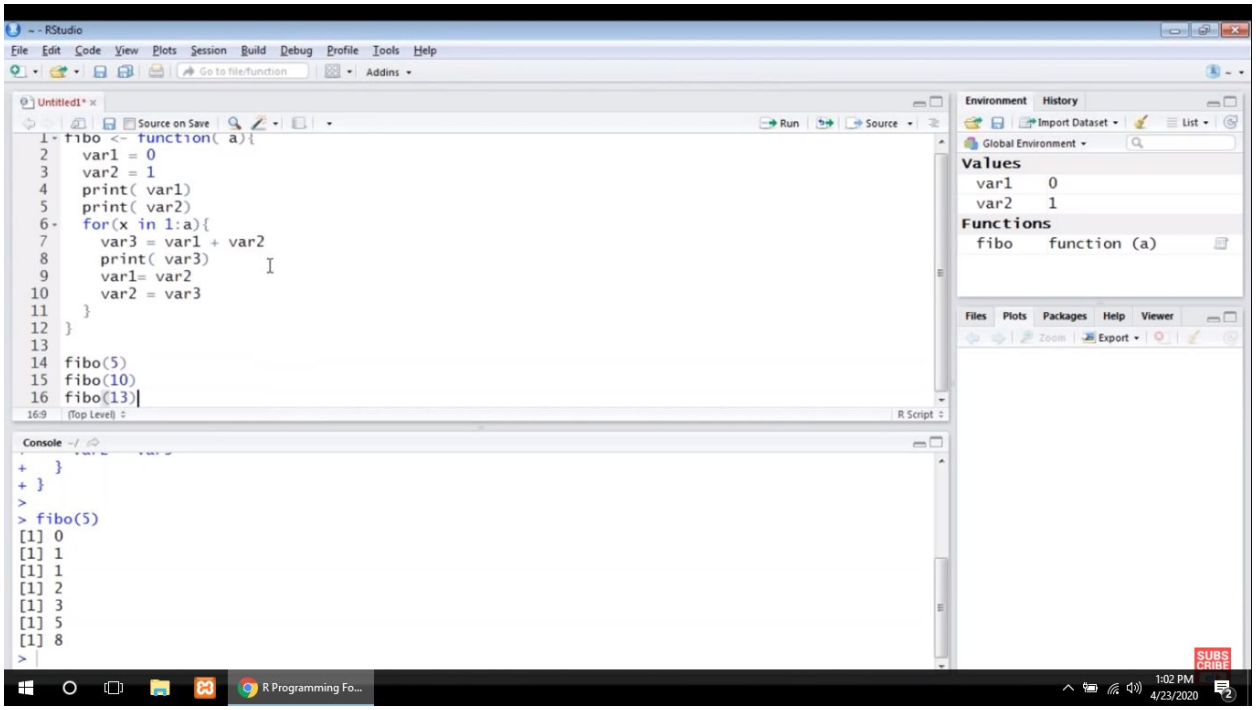

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
1 str1 <- 'Hey, R is fun'
2 print(str1)
3 str2 <- "Hey, R is fun"
4 print(str2)
6 str3 <- paste(str1,str2)
8 print(str3)
Environment History
Import Dataset
Global Environment
Values
str1 "Hey, R is fun"
str2 "Hey, R is fun"
str3 "Hey, R is fun Hey...
var1 24
x 25L
Files Plots Packages Help Viewer
Export
Console
> str1 <- 'Hey, R is fun'
> print(str1)
[1] "Hey, R is fun"
> str2 <- "Hey, R is fun"
> print(str2)
[1] "Hey, R is fun"
> str3 <- paste(str1,str2)
> print(str3)
[1] "Hey, R is fun Hey, R is fun"
R Programming Fo...
12:57 PM
4/23/2020

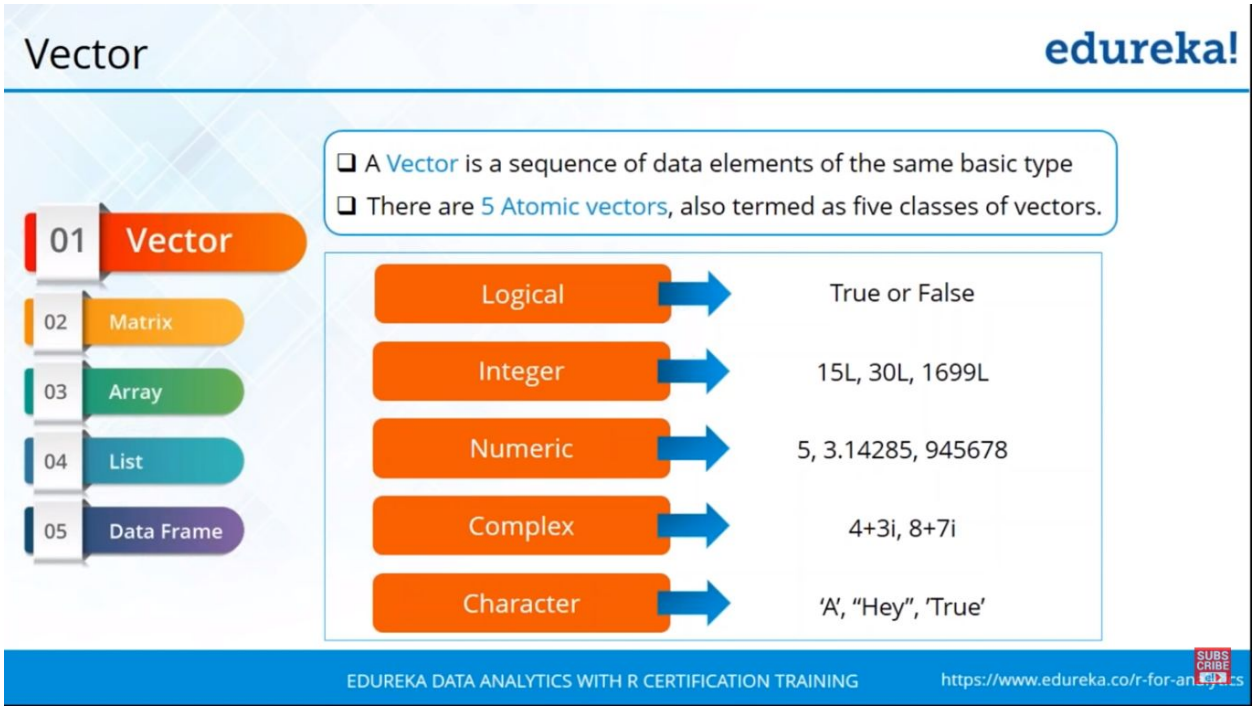

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
1 str1 <- 'Hey, R is fun'
2 print(str1)
3 str2 <- "Hey, R is fun"
4 print(str2)
6 str3 <- paste(str1,str2)
8 print(str3)
10 nchar(str3)
10:1 (Top Level)
R Script
Console ~/
> print(str1)
[1] "Hey, R is fun"
> str2 <- "Hey, R is fun"
> print(str2)
[1] "Hey, R is fun"
> str3 <- paste(str1,str2)
> print(str3)
[1] "Hey, R is fun Hey, R is fun"
> nchar(str3)
[1] 27
Environment History
Import Dataset
List
Global Environment
Values
str1 "Hey, R is fun"
str2 "Hey, R is fun"
str3 "Hey, R is fun Hey...
var1 24
x 25L
Files Plots Packages Help Viewer
Zoom
Export
2 new notifications
R Programming Fo...
12:57 PM
4/23/2020

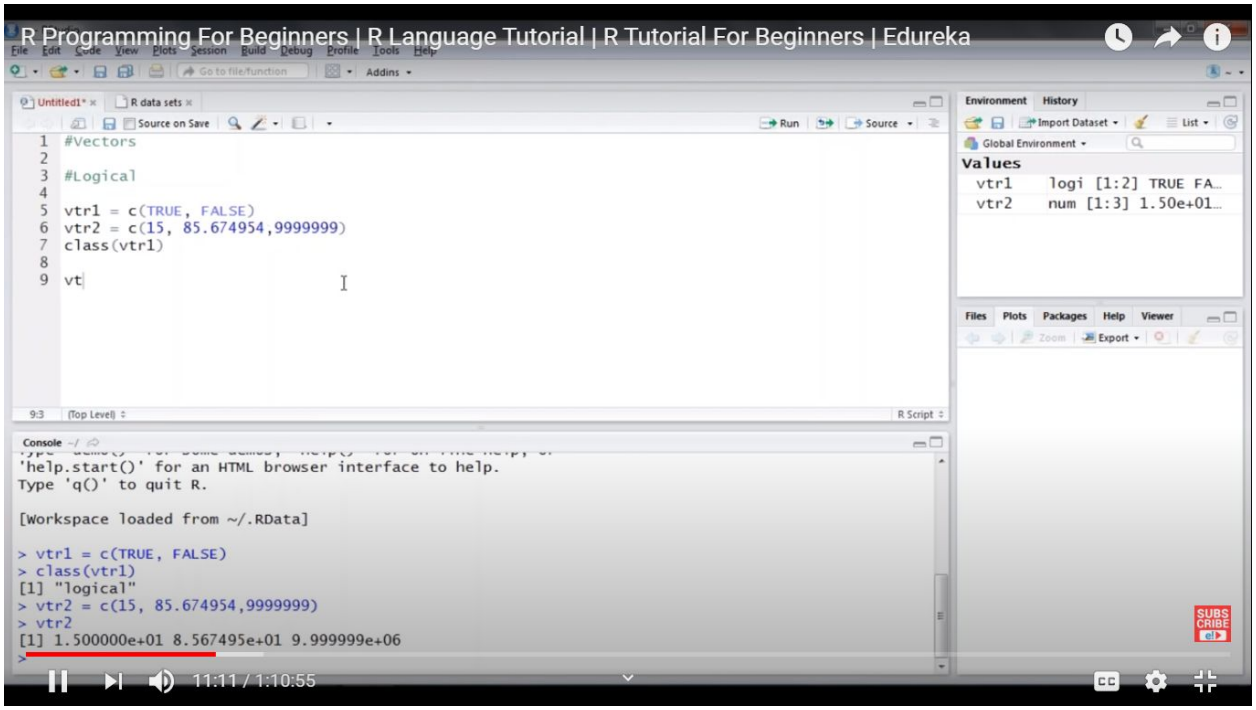

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
3 str2 <- "Hey, R is fun"
4 print(str2)
6 str3 <- paste(str1,str2)
8 print(str3)
10 nchar(str3)
12 str4 = toupper(str1)
13 str5 = tolower(str2)
15 print(str4)
16 print(str5)
12:1 (Top Level)
R Script
Console ~/
> str4 = toupper(str1)
> str5 = tolower(str2)
> print(str4)
[1] "HEY, R IS FUN"
> print(str5)
[1] "hey, r is fun"
Environment History
Import Dataset
List
Global Environment
Values
str1 "Hey, R is fun"
str2 "Hey, R is fun"
str3 "Hey, R is fun Hey...
str4 "HEY, R IS FUN"
str5 "hey, r is fun"
var1 24
Files Plots Packages Help Viewer
Zoom
Export
SUBSCRIBE
R Programming Fo...
12:58 PM
4/23/2020

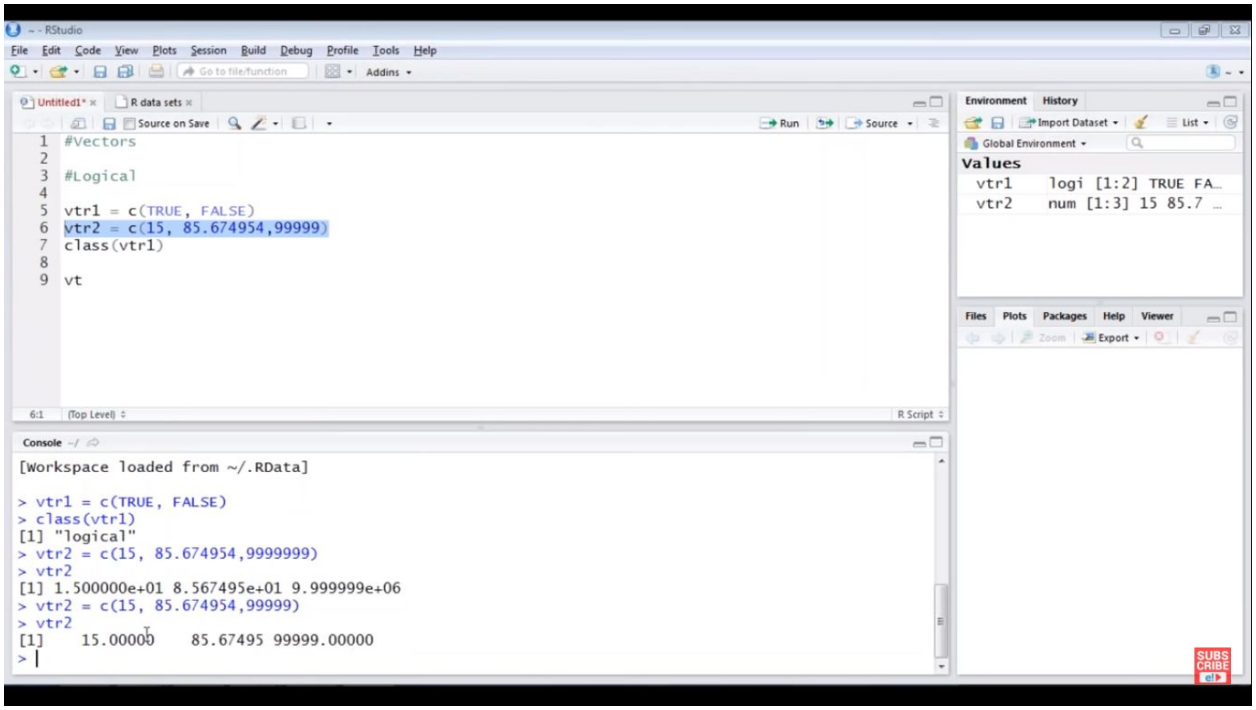

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
4 print(str2)
5
6 str3 <- paste(str1,str2)
7
8 print(str3)
9
10 nchar(str3)
11
12 str4 = toupper(str1)
13 str5 = tolower(str2)
14
15 print(str4)
16 print(str5)
17
18 str6 = substr(str3,5,16)
19 print(str6)
17:1 (Top Level) R Script
Console ~/
> str4 = toupper(str1)
> str5 = tolower(str2)
>
> print(str4)
[1] "HEY, R IS FUN"
> print(str5)
[1] "hey, r is fun"
> str6 = substr(str3,5,16)
> print(str6)
[1] " R is fun He"
>
Environment History
Import Dataset
Global Environment
Values
str1 "Hey, R is fun"
str2 "Hey, R is fun"
str3 "Hey, R is fun Hey...
str4 "HEY, R IS FUN"
str5 "hey, r is fun"
str6 " R is fun He"
Files Plots Packages Help Viewer
Zoom Export
R Programming Fo...
12:59 PM
4/23/2020

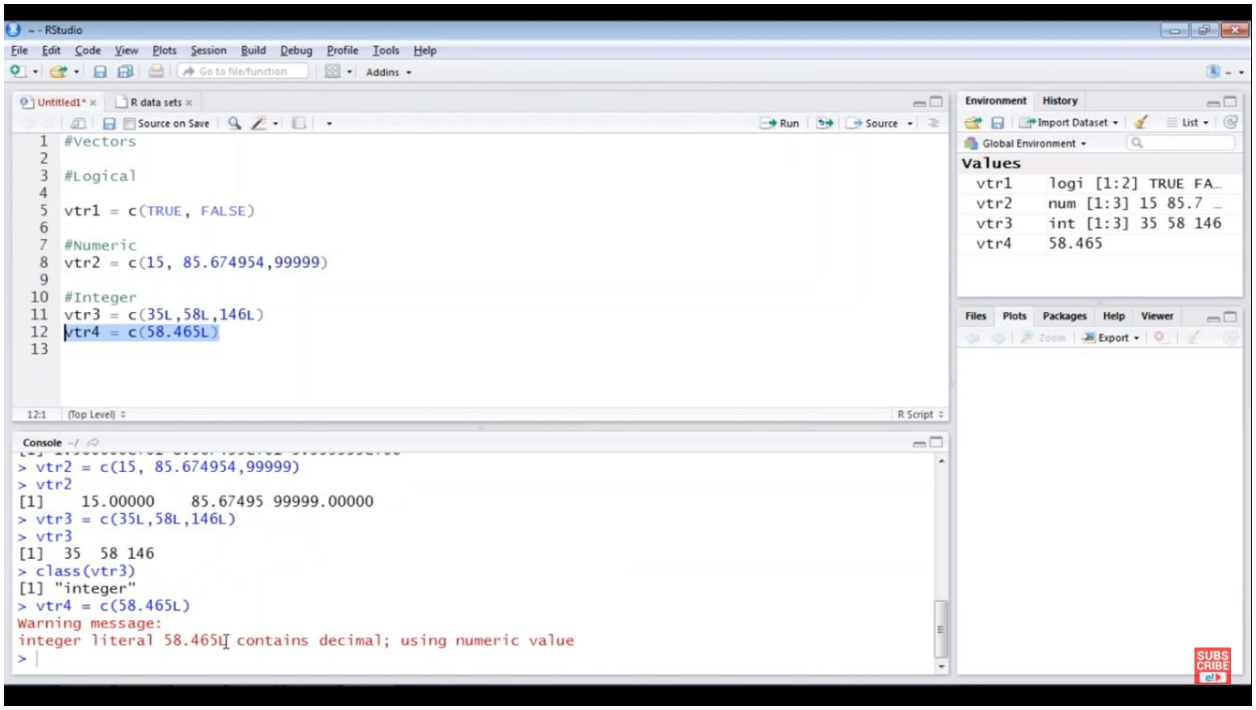

Functions In R
edureka!
Functions
A function is a block of organized, reusable code that is used to perform a single, related action.
Functions
Predefined Functions
User Defined Functions
Function Add
Return 8
Call 1
Call (3,5)
Return 15
Call 2
Call (6,9)
Return 6
Call 3
Call (1,5)
R Programming Fo...
1:00 PM
4/23/2020

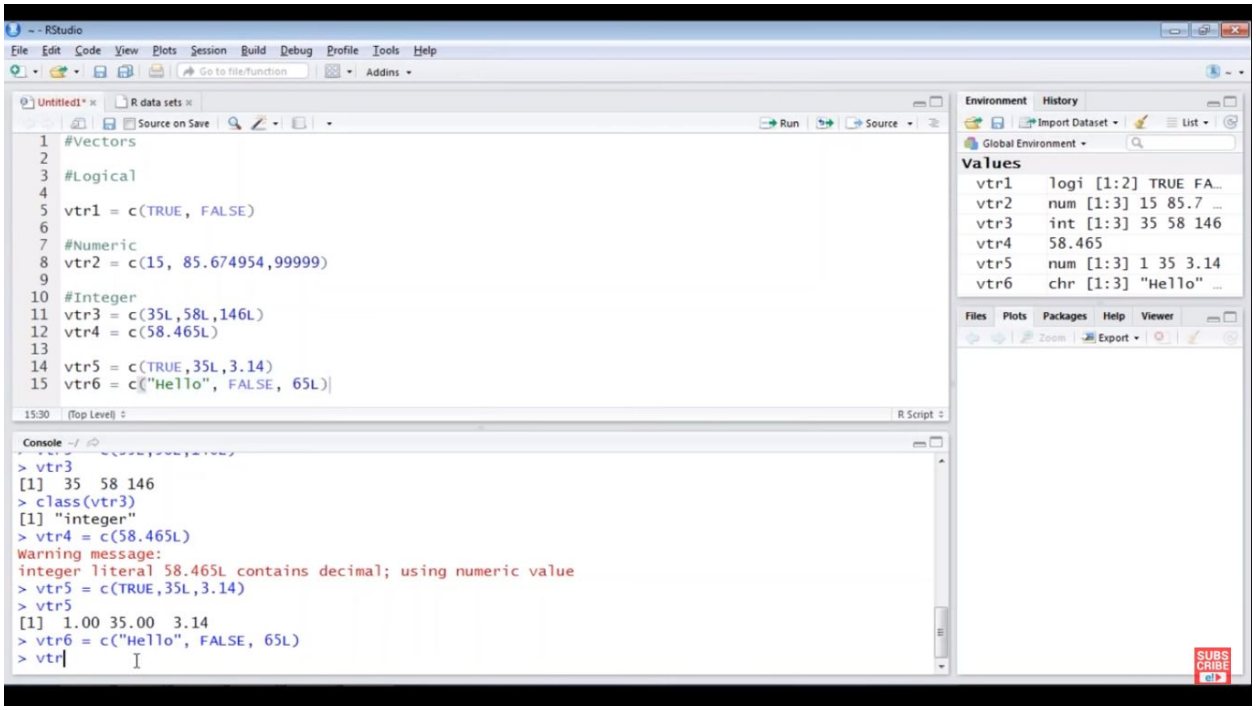

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
fibo <- function(a){
var1 = 0
var2 = 1
print(var1)
print(var2)
for(x in 1:a){
var3 = var1 + var2
print(var3)
var1= var2
var2 = var3
}
}
fibo(5)
fibo(10)
fibo(13)
(Top Level)
R Script
Console ~/
+ }
+ }
> fibo(5)
[1] 0
[1] 1
[1] 1
[1] 2
[1] 3
[1] 5
[1] 8
Environment History
Import Dataset
Global Environment
Values
var1 0
var2 1
Functions
fibo function (a)
Files Plots Packages Help Viewer
Zoom
Export
R Programming Fo...
1:02 PM
4/23/2020

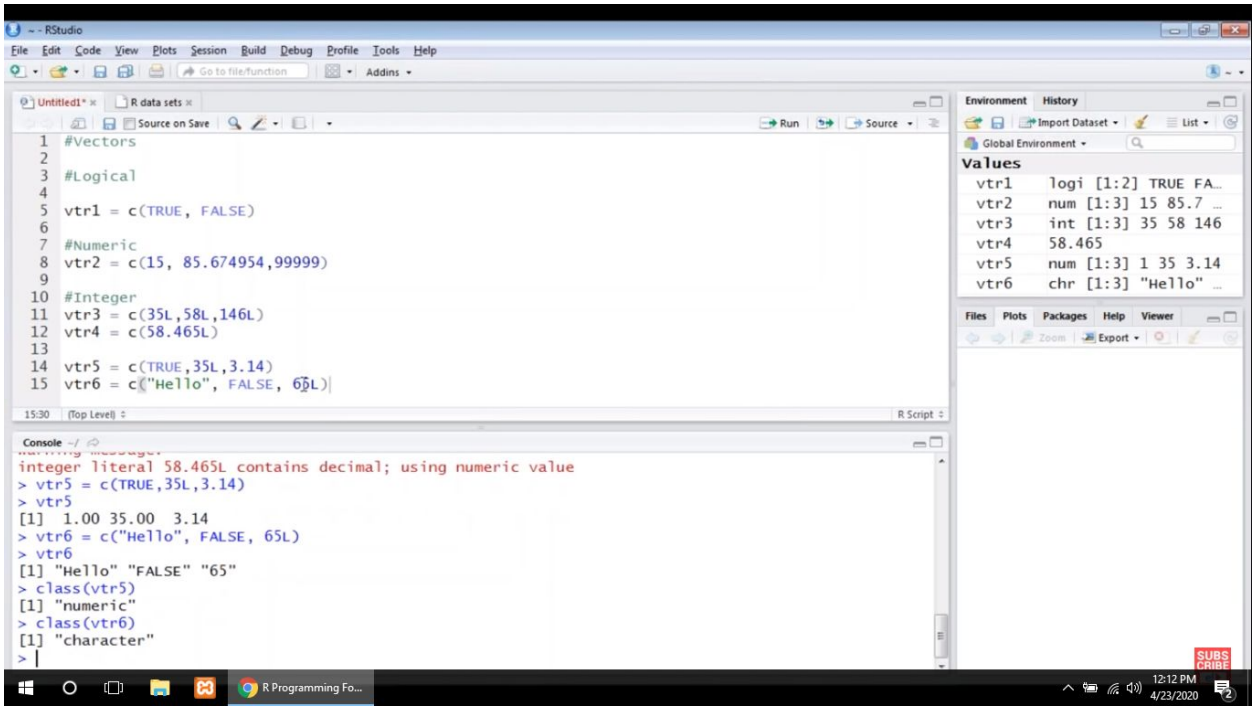

Vector
edureka!
A Vector is a sequence of data elements of the same basic type
There are 5 Atomic vectors, also termed as five classes of vectors.
01 Vector
02 Matrix
03 Array
04 List
05 Data Frame
Logical
True or False
Integer
15L, 30L, 1699L
Numeric
5, 3.14285, 945678
Complex
4+3i, 8+7i
Character
'A', "Hey", 'True'
EDUREKA DATA ANALYTICS WITH R CERTIFICATION TRAINING
https://www.edureka.co/r-for-analytics

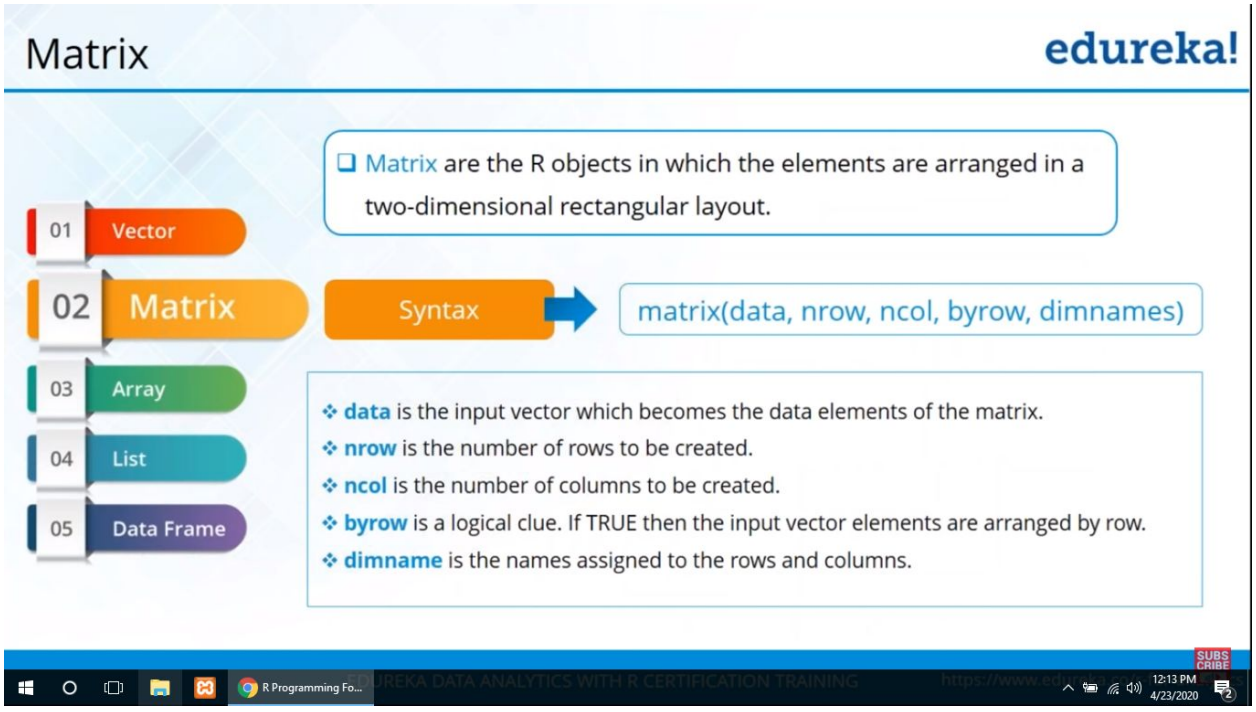

R Programming For Beginners | R Language Tutorial | R Tutorial For Beginners | Edureka
#Vectors
#Logical
vtr1 = c(TRUE, FALSE)
vtr2 = c(15, 85.674954,9999999)
class(vtr1)
vt
'help.start()' for an HTML browser interface to help.
Type 'q()' for quit R.
[Workspace loaded from ~/.RData]
> vtr1 = c(TRUE, FALSE)
> class(vtr1)
[1] "logical"
> vtr2 = c(15, 85.674954,9999999)
> vtr2
[1] 1.500000e+01 8.567495e+01 9.999999e+06
11:11 / 1:10:55

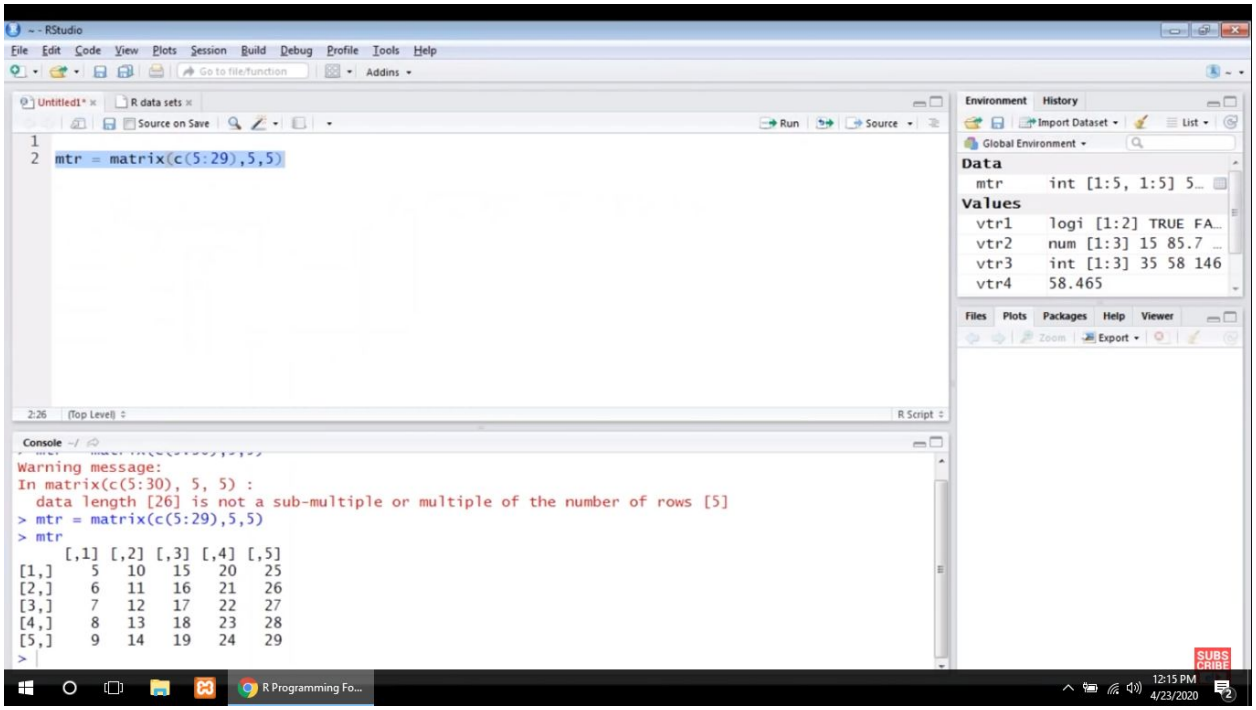

#Vectors
#Logical
vtr1 = c(TRUE, FALSE)
vtr2 = c(15, 85.674954,99999)
class(vtr1)
vt
[Workspace loaded from ~/.RData]
> vtr1 = c(TRUE, FALSE)
> class(vtr1)
[1] "logical"
> vtr2 = c(15, 85.674954,9999999)
> vtr2
[1] 1.500000e+01 8.567495e+01 9.999999e+06
> vtr2 = c(15, 85.674954,99999)
> vtr2
[1] 15.00000 85.67495 99999.00000

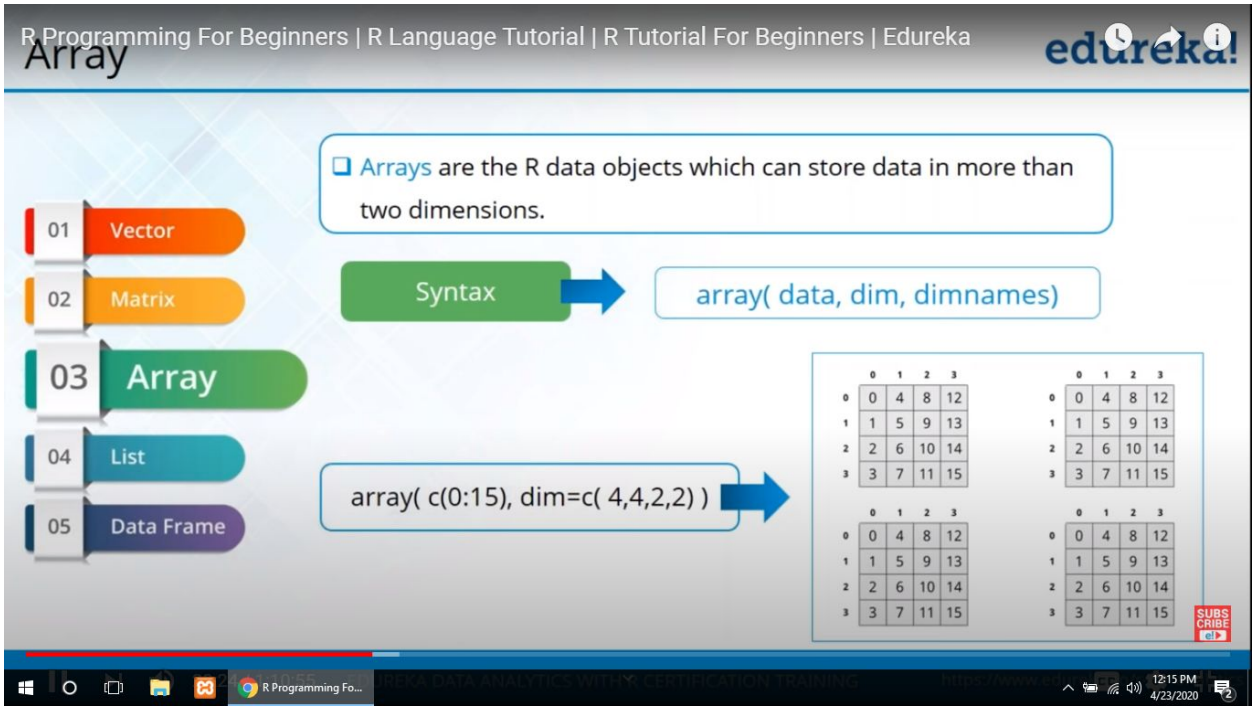

~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1* R data sets
Source on Save Run Source
1 #Vectors
2
3 #Logical
4
5 vtr1 = c(TRUE, FALSE)
6
7 #Numeric
8 vtr2 = c(15, 85.674954,99999)
9
10 #Integer
11 vtr3 = c(35L,58L,146L)
12 vtr4 = c(58.465L)
13
12:1 (Top Level) R Script
Console ~/
> vtr2 = c(15, 85.674954,99999)
> vtr2
[1] 15.00000 85.67495 99999.00000
> vtr3 = c(35L,58L,146L)
> vtr3
[1] 35 58 146
> class(vtr3)
[1] "integer"
> vtr4 = c(58.465L)
Warning message:
integer literal 58.465L contains decimal; using numeric value
>
Environment History
Import Dataset List
Global Environment
Values
vtr1 logi [1:2] TRUE FA…
vtr2 num [1:3] 15 85.7 …
vtr3 int [1:3] 35 58 146
vtr4 58.465
Files Plots Packages Help Viewer
Zoom Export
SUBSCRIBE


~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1* R data sets
Source on Save Run Source
1 #Vectors
2
3 #Logical
4
5 vtr1 = c(TRUE, FALSE)
6
7 #Numeric
8 vtr2 = c(15, 85.674954,99999)
9
10 #Integer
11 vtr3 = c(35L,58L,146L)
12 vtr4 = c(58.465L)
13
14 vtr5 = c(TRUE,35L,3.14)
15 vtr6 = c("Hello", FALSE, 65L)
15:30 (Top Level) R Script
Console ~/
> vtr3
[1] 35 58 146
> class(vtr3)
[1] "integer"
> vtr4 = c(58.465L)
Warning message:
integer literal 58.465L contains decimal; using numeric value
> vtr5 = c(TRUE,35L,3.14)
> vtr5
[1] 1.00 35.00 3.14
> vtr6 = c("Hello", FALSE, 65L)
> vtr
Environment History
Import Dataset List
Global Environment
Values
vtr1 logi [1:2] TRUE FA…
vtr2 num [1:3] 15 85.7 …
vtr3 int [1:3] 35 58 146
vtr4 58.465
vtr5 num [1:3] 1 35 3.14
vtr6 chr [1:3] "Hello" …
Files Plots Packages Help Viewer
Zoom Export
SUBSCRIBE


~ - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1* R data sets
Source on Save Run Source
1 #Vectors
2
3 #Logical
4
5 vtr1 = c(TRUE, FALSE)
6
7 #Numeric
8 vtr2 = c(15, 85.674954,99999)
9
10 #Integer
11 vtr3 = c(35L,58L,146L)
12 vtr4 = c(58.465L)
13
14 vtr5 = c(TRUE,35L,3.14)
15 vtr6 = c("Hello", FALSE, 65L)
15:30 (Top Level) R Script
Console ~/
integer literal 58.465L contains decimal; using numeric value
> vtr5 = c(TRUE,35L,3.14)
> vtr5
[1] 1.00 35.00 3.14
> vtr6 = c("Hello", FALSE, 65L)
> vtr6
[1] "Hello" "FALSE" "65"
> class(vtr5)
[1] "numeric"
> class(vtr6)
[1] "character"
>
Environment History
Import Dataset List
Global Environment
Values
vtr1 logi [1:2] TRUE FA…
vtr2 num [1:3] 15 85.7 …
vtr3 int [1:3] 35 58 146
vtr4 58.465
vtr5 num [1:3] 1 35 3.14
vtr6 chr [1:3] "Hello" …
Files Plots Packages Help Viewer
Zoom Export
SUBSCRIBE
R Programming Fo...
12:12 PM
4/23/2020


Matrix
edureka!
Matrix are the R objects in which the elements are arranged in a two-dimensional rectangular layout.
01 Vector
02 Matrix
03 Array
04 List
05 Data Frame
Syntax
matrix(data, nrow, ncol, byrow, dimnames)
data is the input vector which becomes the data elements of the matrix.
nrow is the number of rows to be created.
ncol is the number of columns to be created.
byrow is a logical clue. If TRUE then the input vector elements are arranged by row.
dimname is the names assigned to the rows and columns.
SUBSCRIBE
R Programming Fo...
EDUREKA DATA ANALYTICS WITH R CERTIFICATION TRAINING
12:13 PM
4/23/2020


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1* R data sets
Source on Save Run Source
1 mtr = matrix(c(5:29),5,5)
2:26 (Top Level) R Script
Console ~/
Warning message:
In matrix(c(5:30), 5, 5) :
data length [26] is not a sub-multiple or multiple of the number of rows [5]
> mtr = matrix(c(5:29),5,5)
> mtr
[,1] [,2] [,3] [,4] [,5]
[1,] 5 10 15 20 25
[2,] 6 11 16 21 26
[3,] 7 12 17 22 27
[4,] 8 13 18 23 28
[5,] 9 14 19 24 29
Environment History
Import Dataset List
Global Environment
Data
mtr int [1:5, 1:5] 5…
Values
vtr1 logi [1:2] TRUE FA…
vtr2 num [1:3] 15 85.7 …
vtr3 int [1:3] 35 58 146
vtr4 58.465
Files Plots Packages Help Viewer
Zoom Export
R Programming Fo…
12:15 PM
4/23/2020


R Programming For Beginners | R Language Tutorial | R Tutorial For Beginners | Edureka
edureka!
Array
Arrays are the R data objects which can store data in more than two dimensions.
01 Vector
02 Matrix
03 Array
04 List
05 Data Frame
Syntax
array(data, dim, dimnames)
array(c(0:15), dim=c(4,4,2,2))
R Programming Fo…
12:15 PM
4/23/2020