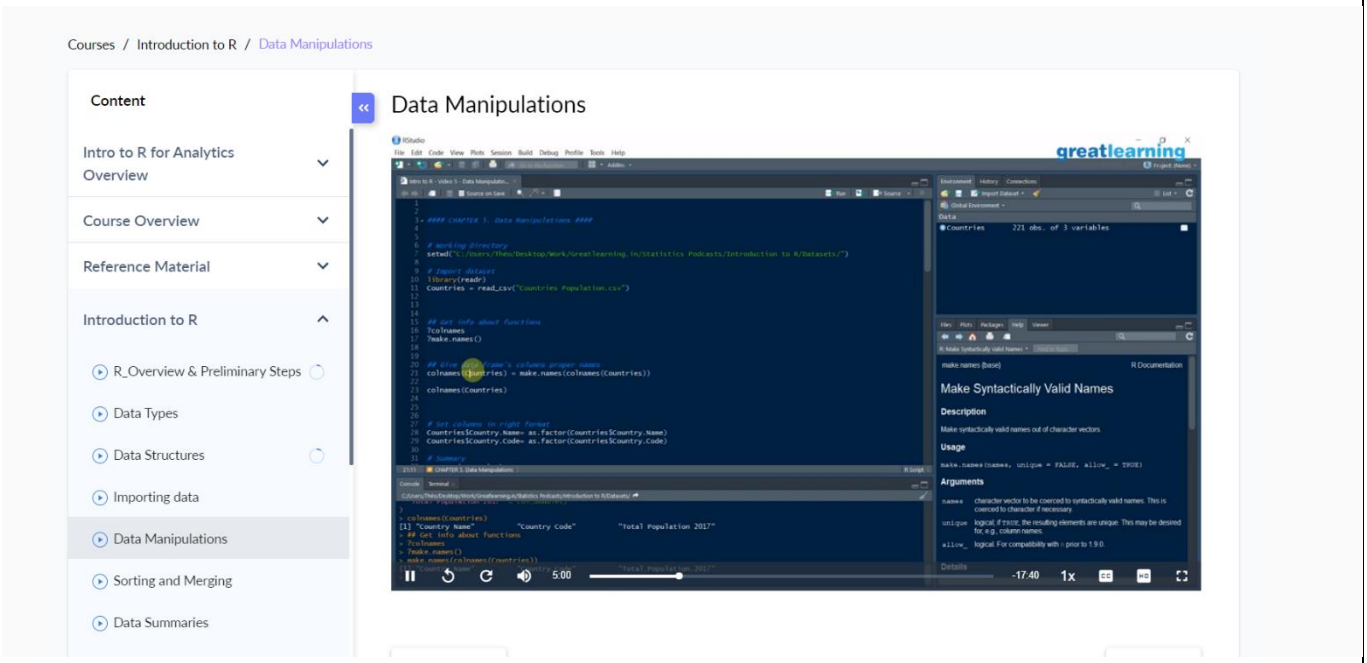
DAILY ASSESSMENT FORMAT

Date:	**11 JUNE 2020**	**Name:**	**PAVITHRAN S**
Course:	**R PROGRAMMING**	**USN:**	**4AL17EC068**
Topic:	**R PROGRAM MANIPULATION**	**Semester & Section:**	**6TH B**
Github Repository:	**Pavithran**		

FORENOON SESSION DETAILS

Image of session



Report – Report can be typed or hand written for up to two pages.

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. R language is rich in built-in operators and provides following types of operators.

Types of Operators

We have the following types of operators in R programming −

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Assignment Operators
- Miscellaneous Operators

Arithmetic Operators

Following table shows the arithmetic operators supported by R language. The operators act on each element of the vector.

Operator	Description	Example

+	Adds two vectors	Live De
		`v <- c( 2,5.5,6)`
		`t <- c(8, 3, 4)`
		`print(v+t)`
		it produces the following result −
		`[1] 10.0 8.5 10.0`

−	Subtracts second vector from the first	Live De
		`v <- c( 2,5.5,6)`
		`t <- c(8, 3, 4)`
		`print(v-t)`
		it produces the following result −
		`[1] -6.0 2.5 2.0`

*	Multiplies both vectors	Live De
		`v <- c( 2,5.5,6)`
		`t <- c(8, 3, 4)`
		`print(v*t)`
		it produces the following result −
		`[1] 16.0 16.5 24.0`

/	Divide the first vector with the second	Live De
		`v <- c( 2,5.5,6)`
		`t <- c(8, 3, 4)`
		`print(v/t)`
		When we execute the above code, it produ
		the following result −
		`[1] 0.250000 1.833333 1.500000`

%%	Give the remainder of the first vector with the second	Live De
		`v <- c( 2,5.5,6)`
		`t <- c(8, 3, 4)`
		`print(v%%t)`
		it produces the following result −

		`[1] 2.0 2.5 2.0`

%/%	The result of division of first vector with second (quotient)	Live Demo
		`v <- c( 2,5.5,6)`
		`t <- c(8, 3, 4)`
		`print(v%/%t)`
		it produces the following result −
		`[1] 0 1 1`

^	The first vector raised to the exponent of second vector	Live Demo
		`v <- c( 2,5.5,6)`
		`t <- c(8, 3, 4)`
		`print(v^t)`
		it produces the following result −
		`[1] 256.000 166.375 1296.000`

Relational Operators

Following table shows the relational operators supported by R language. Each element of the first vector is compared with the corresponding element of the second vector. The result of comparison is a Boolean value.

Operator	Description	Example

>	Checks if each element of the first vector is greater than the corresponding element of the second vector.	`v <- c(2,5.5,6,9)`
		`t <- c(8,2.5,14,9)`
		`print(v>t)`
		it produces the following result −
		`[1] FALSE TRUE FALSE FALSE`

<	Checks if each element of the first vector is less than the corresponding element of the second vector.	`v <- c(2,5.5,6,9)`
		`t <- c(8,2.5,14,9)`
		`print(v < t)`
		it produces the following result −
		`[1] TRUE FALSE TRUE FALSE`

==	Checks if each element of the first vector is equal to the corresponding element of the second vector.	`v <- c(2,5.5,6,9)`
		`t <- c(8,2.5,14,9)`
		`print(v == t)`
		it produces the following result −

		`[1] FALSE FALSE FALSE TRUE`

<=	Checks if each element of the first vector is less than or equal to the corresponding element of the second vector.	`v <- c(2,5.5,6,9)`
		`t <- c(8,2.5,14,9)`
		`print(v<=t)`
		it produces the following result −
		`[1] TRUE FALSE TRUE TRUE`

>=	Checks if each element of the first vector is greater than or equal to the corresponding element of the second vector.	`v <- c(2,5.5,6,9)`
		`t <- c(8,2.5,14,9)`
		`print(v>=t)`
		it produces the following result −
		`[1] FALSE TRUE FALSE TRUE`

!=	Checks if each element of the first vector is unequal to the corresponding element of the second vector.	`v <- c(2,5.5,6,9)`
		`t <- c(8,2.5,14,9)`
		`print(v!=t)`
		it produces the following result −
		`[1] TRUE TRUE TRUE FALSE`

Logical Operators

Following table shows the logical operators supported by R language. It is applicable only to vectors of type logical, numeric or complex. All numbers greater than 1 are considered as logical value TRUE.

Each element of the first vector is compared with the corresponding element of the second vector. The result of comparison is a Boolean value.

Operator	Description	Example

&	It is called Element-wise Logical AND operator. It combines each element of the first vector with the corresponding element of the second vector and gives a output TRUE if both the elements are TRUE.	`v <- c(3,1,TRUE,2+3i)`
		`t <- c(4,1,FALSE,2+3i)`
		`print(v&t)`
		it produces the following result −
		`[1] TRUE TRUE FALSE TRUE`

|	It is called Element-wise Logical OR operator. It combines each element of the first vector with the corresponding element of the second vector and gives a output TRUE if one the elements is TRUE.	`v <- c(3,0,TRUE,2+2i)`
		`t <- c(4,0,FALSE,2+3i)`
		`print(v|t)`
		it produces the following result −
		`[1] TRUE FALSE TRUE TRUE`

!	It is called Logical NOT operator. Takes each element of the vector and gives the opposite logical value.	`v <- c(3,0,TRUE,2+2i)`
		`print(!v)`
		it produces the following result −
		`[1] FALSE TRUE FALSE FALSE`

The logical operator && and || considers only the first element of the vectors and give a vector of single element as output.

Operator	Description	Example

&&	Called Logical AND operator. Takes first element of both the vectors and gives the TRUE only if both are TRUE.	`v <- c(3,0,TRUE,2+2i)`
		`t <- c(1,3,TRUE,2+3i)`
		`print(v&&t)`
		it produces the following result −
		`[1] TRUE`

||	Called Logical OR operator. Takes first element of both the vectors and gives the TRUE if one of them is TRUE.	`v <- c(0,0,TRUE,2+2i)`
		`t <- c(0,3,TRUE,2+3i)`
		`print(v||t)`
		it produces the following result −
		`[1] FALSE`

Assignment Operators

These operators are used to assign values to vectors.

Operator	Description	Example

<−
or
	Called Left Assignment	`v1 <- c(3,1,TRUE,2+3i)`
		`v2 <<- c(3,1,TRUE,2+3i)`
		`v3 = c(3,1,TRUE,2+3i)`
		`print(v1)`

= or <<–		print(v2) print(v3) it produces the following result − [1] 3+0i 1+0i 1+0i 2+3i [1] 3+0i 1+0i 1+0i 2+3i [1] 3+0i 1+0i 1+0i 2+3i
 ->> or ->>	Called Right Assignment	c(3,1,TRUE,2+3i) -> v1 c(3,1,TRUE,2+3i) ->> v2 print(v1) print(v2) it produces the following result − [1] 3+0i 1+0i 1+0i 2+3i [1] 3+0i 1+0i 1+0i 2+3i