**+ Adds two vectors**. **- Subtracts second vector from the first**

v <- c( 2,5.5,6)

t <- c(8, 3, 4)

print(v+t)

\* Multiplies both vectors / Divide the first vector with the second

p <- c (4,5.56,6) p <- c (8,5.5,7)

q <- c (8, 7, 4) q <- c (8, 3, 4)

print(p/q)

print(p\*q)

**%% Give the remainder of the first vector with the second**

v <- c( 2,5.5,6)

w<- c(8, 3, 4)

print(v%%w)

**%/% The result of division of first vector with second (quotient)**

v <- c(1,5,6) w<- c(8, 3, 4)

print(v%/%w)

**: Colon operator. It creates the series of numbers in sequence for a vector**.

s <- 1:10

print(s)

**%in% This operator is used to identify if an element belongs to a vector.**

a <- 8

b <- 12

c <- 1:10

print(a %in% t) print(b %in% t)

**%\*% This operator is used to multiply a matrix with its transpose**

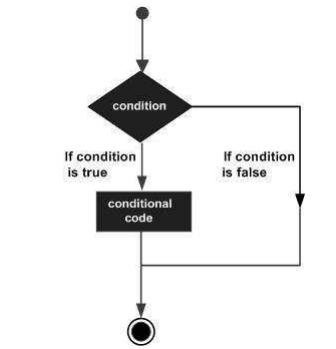
M = matrix( c(1,1,0,0), nrow=2,ncol=2,byrow = TRUE)

t = M %\*% t(M)

print(t).

**Control structures**

Decision making structures require the programmer to specify one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

****

IF statement

An if statement consists of a Boolean expression followed by one or more statements.

if(boolean\_expression)

{

// statement(s) will execute if the boolean expression is true.

}

-----------------------------------------------------

X <-8

if(is.integer(x))

{

print("X is an Integer")

}

-------------------------------------------------------------

if (2 > 0)

{

print("true")

}

An **if** statement can be followed by an optional **else** statement which executes when the boolean expression is false.

The basic syntax for creating an if...else statement in R is:

if(Boolean expression) {

// statement(s) will execute if the Boolean expression is true.

} else

{

// statement(s) will execute if the Boolean expression is false.

}

x <- c("what","is","truth")

if("Truth" %in% x)

{ print("Truth is found")

} else

{

print("Truth is not found")

}

**The if...else if...else Statement**

An if statement can be followed by an optional else if...else statement, which is very useful to test various conditions using single if...else if statement.

When using if, else if, else statements there are few points to keep in mind.

• An if can have zero or one else and it must come after any else if's.

• An if can have zero to many else if's and they must come before the else.

**The basic syntax for creating an if...else if...else statement in R is:**

if(boolean\_expression 1)

{ // Executes when the boolean expression 1 is true. }

else if( boolean\_expression 2)

{ // Executes when the boolean expression 2 is true. }

else if( boolean\_expression 3)

{ // Executes when the boolean expression 3 is true. }

else { // executes when none of the above condition is true.

}

x <- c("what","is","truth")

if("Truth" %in% x)

{ print("Truth is found the first time") }

else if ("truth" %in% x)

{ print("truth is found the second time") }

else { print("No truth found")

}

**Loops**

**While Loops**

# Set i equal to 1

i <- 1

# While i is less than or equal to three, print i

# The loop will increment the value of i after each print

while (i <= 3)

{ print(i)

i <- i + 1

}

**R -Repeat Loop**

The Repeat loop executes the same code again and again until a stop condition is met.

The basic syntax for creating a repeat loop in R is:

repeat {

commands

if(condition)

{ break

v <- c("Hello","loop")

count <- 1

repeat{

print(v)

count <- count+1

if(count > 5)

{ break }

}

**R For loop**

for loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

The basic syntax for creating a for loop statement in R is:

for (value in vector) { statements }

v <- LETTERS[1:4]

for ( i in v)

{

print(i)

}

**Loop Control Statements**

Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed.

**Break** : Terminates the loop statement and transfers execution to the statement immediately following the loop.

The break statement in R programming language has the following two usages:

• When the break statement is encountered inside a loop, the loop is immediately terminated, and program control resumes at the next statement following the loop.

• It can be used to terminate a case in the switch statement.

**The basic syntax for creating a break statement in R is: break**

**The next statement in R** programming language is useful when we want to skip the current iteration of a loop without terminating it. On encountering next, the R parser skips further evaluation and starts next iteration of the loop.

**The basic syntax for creating a next statement in R is: next**

v <- LETTERS[1:6]

for ( i in v)

{

if (i == "D")

{ next }

print(i)

}

employees <- list("jane", "john")

for (employee in employees)

{

print(employee)

}

x <- 20

if (x < 20)

{ print("x is less than 20") }

else if (x > 20)

{ print("x is greater than 20") }

else { print("x is equal to 20") }

data()