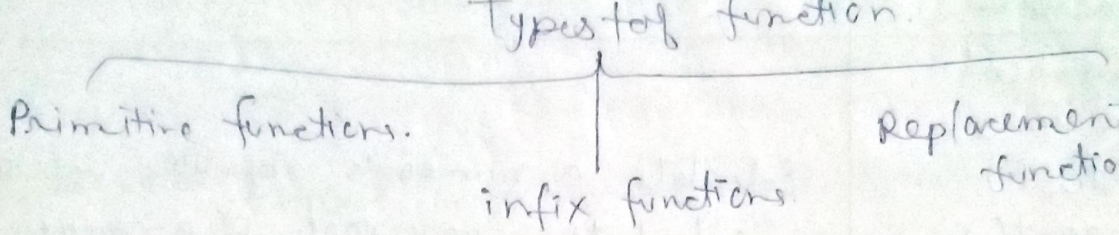
Built in Function in R

Functions	
Mathematical Functions	
(i) abs()	calculate a number's absolute value
(ii) sqrt()	calculate square root of a number
(iii) round()	round a number to nearest integer
(iv) exp()	calculate the exponential value
(v) log()	calculate logarithmic value of number
(vi) cos(), sin(), tan()	calculate the trigonometric value.
Statistical function.	
(i) mean()	vectors arithmetic mean/average.
(ii) median()	vector's median value
(iii) cor()	correlation between two vector
(iv) var()	variance of the vector
Data manipulation functions	
(i) unique()	return unique value in vector
(ii) subset()	subset a data frame based on conditions
(iii) aggregate()	groups data according to a grouping variable
(iv) order()	uses ascending or descending order to sort a vector
file input/output functions	
(i) read.csv()	reads information from a csv file.
(ii) file.csv()	publishes information to a read csv file
(iii) file.table()	reads information from a tabular.
(iv) file table()	creates a tabular file with data

TYPES OF FUNCTIONS IN R



(i) Primitive functions:
Generally, a function comprises of three parts.

- The formals(), the list of arguments that control how you call the function.
- The body(), the code inside the function
- The environment(), the data structure that determines how the function find the values associated with the names.

- The formals and body are defined explicitly whenever one creates a function, but the environment is specified implicitly, based on where you define the function.
- But there is an exception to the rule that a function has three components, some functions call C code directly. These functions as known as primitive functions.
- The primitive functions exist primarily in C not R, so their formals(), body() and environment() are NULL
- These functions are only found in base package
- primitive functions are harder to write but are highly efficient. They are of two types either type builtin or type special.

(ii) Infix functions:

- infix function are those function in which the function name comes in between its arguments. and hence have two arguments.
- R comes with a number of built-in infix operator such as: , ::, :::, $, @, ^, *, /, +, -, >, <, =, <=, >=, ==, !=, !, &, |, ||, ~, <-, and <<-
- coder can create his own infix functions that start and end with %.
- The names of an infix function is more flexible. as it can contain any sequence of character except %. There are some predefined infix operators in R programming.

Operator	Description
%%	Remainder operator
%/%	integer Division
%*%	matrix multiplication
%o%	outer product
%x%	kronecker product
%in%	matching operator.

(iii) Replacement Functions:

- Replacement functions modify their arguments inplace
- The name of replacement functions are always succeeded by <-
- They must have arguments named x and value. return the modified object. In case of a replacement, a function needs additional arguments, the additional arguments should be placed between x and value, and must be called with additional arguments on the left.

- The name of the function has to be quoted as it is a syntactically valid but non-standard name and the parser would interpret <- as the operator not as part of the function name if it weren't quoted.

Syntax:

```
"function_name<-"<- function(x, additional arguments, value)
{
  function body
}
```

example

```
"replace<-"<- function(x, value)
{
    x[1] = value
    x
}
x = rep.int(5,7)
replace(x) = 0L
print(x)
```