R quick reference guide

Creating a vector

```
x <- 10
```

a <- c(2 , 3 , 4)	2 3 4
a <- 2 : 6	2 3 4 5 6
a <- seq(2, 3, by=0.5)	2.0 2.5 3.0
a <- rep(1:2, times=3)	1 2 1 2 1 2
a <- rep(1:2, each=3)	1 1 1 2 2 2

Vector Functions

sort(x)
Return x sorted.

rev(x)
Return x reversed.

table(x)
See counts of values.

unique(x)
See unique values.

Using Libraries

```
library(dplyr)
# including the package during the session

dplyr::select
# include any particular function from the package

data(iris)
# loading a built-in dataset into the enviornment
```

Selecting Vector Elements

By Position

x[4]	The fourth element.
x[-4]	All but the fourth.
x[2:4]	Elements two to four.
x[-(2:4)]	All elements except two to four.
x[c(1, 5)]	Elements one and five.

By Value

x[x == 10]	Elements which are equal to 10.
x[x < 0]	All elements less than zero.
x[x %in% c(1, 2, 5)]	Elements in the set 1, 2, 5.

Named Vectors

x['apple']	Element with name 'apple'.

For loop

```
a <- 0
for (i in 0:10){
  a <- a + 1
}
```

While loop

```
i <- 0
while (i < 5){
  i <- i + 1
}
```

If loop

```
if (i > 3){
  print('Yes')
} else {
  print('No')
}
```

Function

```
square <- function(x){
  squared <- x*x
  return(squared)
}
```

Read and write to file

df <- read.table('file.txt')	write.table(df, 'file.txt')	Read and write a delimited text file.
df <- read.csv('file.csv')	write.csv(df, 'file.csv')	Read and write a comma separated value file. This is a special case of read.table/ write.table.
load('file.RData')	save(df, file = 'file.Rdata')	Read and write an R data file, a file type special for R.

Matrices

```
m <- matrix(x, nrow = 3, ncol = 3)
```
Create a matrix from x.


`m[2, ]` - Select a row


`m[ , 1]` - Select a column


`m[2, 3]` - Select an element

`t(m)`
Transpose

`m %*% n`
Matrix Multiplication

`solve(m, n)`
Find x in: m * x = n

Lists

```
l <- list(x = 1:5)
# create a list of vectors from 1 to 5
```

`l[[2]]`	`l[1]`	`l$x`	`l['y']`
Second element of l.	New list with only the first element.	Element named x.	New list with only element named y.

Dataframe

```
df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))
```
A special case of a list where all elements are the same length.

x	y
1	a
2	b
3	c

List subsetting

`df$x` 

`df[[2]]` 

Understanding a data frame

`View(df)`	See the full data frame.
`head(df)`	See the first 6 rows.

Matrix subsetting

`df[ , 2]` 

`df[2, ]` 

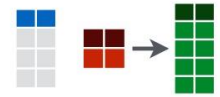
`df[2, 2]` 

`nrow(df)` Number of rows.

`ncol(df)` Number of columns.

`dim(df)` Number of columns and rows.

`cbind` - Bind columns. 

`rbind` - Bind rows. 

Mathematical Functions

`log(x)`	Natural log.	`sum(x)`	Sum.
`exp(x)`	Exponential.	`mean(x)`	Mean.
`max(x)`	Largest element.	`median(x)`	Median.
`min(x)`	Smallest element.	`quantile(x)`	Percentage quantiles.
`round(x, n)`	Round to n decimal places.	`rank(x)`	Rank of elements.
`signif(x, n)`	Round to n significant figures.	`var(x)`	The variance.
`cor(x, y)`	Correlation.	`sd(x)`	The standard deviation.

Statistics

`lm(x ~ y, data=df)`
Linear model.

`glm(x ~ y, data=df)`
Generalised linear model.

`summary`
Get more detailed information out a model.

`t.test(x, y)`
Preform a t-test for difference between means.

`pairwise.t.test`
Preform a t-test for paired data.

`prop.test`
Test for a difference between proportions.

`aov`
Analysis of variance.

```
attach(df_name)
# this adds the dataframe or the list to R's search path so if you want to access a column
# of the dataframe you don't need to do mean(df$column_name) ,  you do mean(column_name)
```

Conditions

`a == b`	Are equal	`a > b`	Greater than	`a >= b`	Greater than or equal to	`is.na(a)`	Is missing
`a != b`	Not equal	`a < b`	Less than	`a <= b`	Less than or equal to	`is.null(a)`	Is null

```
names() # This method in R can be used in two ways one to set the names of the objects.
# second to get the name of the objects.
names(x) <- ("name1" , "name2")
names(df) # returns the column names of the dataframe.
```