

# Base R

## Cheat Sheet

### Getting Help

Accessing the help files

**?mean**

Get help of a particular function.

**help.search('weighted mean')**

Search the help files for a word or phrase.

**help(package = 'dplyr')**

Find help for a package.

More about an object

**str(iris)**

Get a summary of an object's structure.

**class(iris)**

Find the class an object belongs to.

### Using Packages

**install.packages('dplyr')**

Download and install a package from CRAN.

**library(dplyr)**

Load the package into the session, making all its functions available to use.

**dplyr::select**

Use a particular function from a package.

**data(iris)**

Load a built-in dataset into the environment.

### Working Directory

**getwd()**

Find the current working directory (where inputs are found and outputs are sent).

**setwd('C://file/path')**

Change the current working directory.

**Use projects in RStudio to set the working directory to the folder you are working in.**

### Vectors

#### Creating Vectors

c(2, 4, 6)	2 4 6	Join elements into a vector
2:6	2 3 4 5 6	An integer sequence
seq(2, 3, by=0.5)	2.0 2.5 3.0	A complex sequence
rep(1:2, times=3)	1 2 1 2 1 2	Repeat a vector
rep(1:2, each=3)	1 1 1 2 2 2	Repeat elements of a vector

#### Vector Functions

**sort(x)**

Return x sorted.

**table(x)**

See counts of values.

**rev(x)**

Return x reversed.

**unique(x)**

See unique values.

#### 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'.

### Programming

#### For Loop

**for (variable in sequence) {**

**Do something**

**}**

##### Example

```
for (i in 1:4){
  j <- i + 10
  print(j)
}
```

#### While Loop

**while (condition) {**

**Do something**

**}**

##### Example

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

#### If Statements

**if (condition) {**

**Do something**

**} else {**

**Do something different**

**}**

##### Example

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

#### Functions

**function\_name <- function(var) {**

**Do something**

**return(new\_variable)**

**}**

##### Example

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

### Reading and Writing Data

Also see the **readr** package.

Input	Output	Description
<b>df &lt;- read.table('file.txt')</b>	<b>write.table(df, 'file.txt')</b>	Read and write a delimited text file.
<b>df &lt;- read.csv('file.csv')</b>	<b>write.csv(df, 'file.csv')</b>	Read and write a comma separated value file. This is a special case of readtable/writeable.
<b>load('file.Rdata')</b>	<b>save(df, file = 'file.Rdata')</b>	Read and write an R data file, a file type special for R.

#### Conditions

<b>a == b</b>	<b>Are equal</b>	<b>a &gt; b</b>	<b>Greater than</b>	<b>a &gt;= b</b>	<b>Greater than or equal to</b>	<b>is.na(a)</b>	<b>Is missing</b>
<b>a != b</b>	<b>Not equal</b>	<b>a &lt; b</b>	<b>Less than</b>	<b>a &lt;= b</b>	<b>Less than or equal to</b>	<b>is.null(a)</b>	<b>Is null</b>

## Types

Converting between common data types in R. Can always go from a higher value in the table to a lower value.

as.logical	TRUE, FALSE, TRUE	Boolean values (TRUE or FALSE).
as.numeric	1, 0, 1	Integers or floating point numbers.
as.character	'1', '0', '1'	Character strings. Generally preferred to factors.
as.factor	'1', '0', '1', levels: '1', '0'	Character strings with preset levels. Needed for some statistical models.

## Maths Functions

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

## Variable Assignment

```
> a <- 'apple'
> a
[1] 'apple'
```

## The Environment

<b>ls()</b>	List all variables in the environment.
<b>rm(x)</b>	Remove x from the environment.
<b>rm(list = ls())</b>	Remove all variables from the environment.

**You can use the environment panel in RStudio to browse variables in your environment.**

## Matrices

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

 <b>m[2, ]</b>	- Select a row
 <b>m[, 1]</b>	- Select a column
 <b>m[2, 3]</b>	- Select an element

**t(m)**  
Transpose  
 $m \%*\% n$   
Matrix Multiplication  
**solve(m, n)**  
Find x in:  $m \cdot x = n$

## Lists

```
l <- list(x = 1:5, y = c('a', 'b'))
A list is a collection of elements which can be of different types.
```



<b>l[[2]]</b>	<b>l[1]</b>	<b>l\$x</b>	<b>l[["y"]]</b>
Second element only of l.	New list with only the first element.	Element named x.	New list with only element named y.



Also see the **dplyr** package.

## Data Frames

```
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




**df\$x**  
  
**df[[2]]**  



**df\$y**  
  
**df[[1]]**  



### List subsetting

*Understanding a data frame*  
See the full data frame.  
**view(df)**  
See the first 6 rows.

### Matrix subsetting

<b>df[, 2]</b>	 Number of rows.	<b>nrow(df)</b>
<b>df[2, ]</b>	 Number of columns.	<b>ncol(df)</b>
<b>df[2, 1]</b>	 Number of columns and rows.	<b>dim(df)</b>

**cbind** - Bind columns.  


**rbind** - Bind rows.  


## Strings

Also see the **stringr** package.

<b>paste(x, y, sep = ' ')</b>	Join multiple vectors together.
<b>paste(x, collapse = ' ')</b>	Join elements of a vector together.
<b>grep(pattern, x)</b>	Find regular expression matches in x.
<b>gsub(pattern, replace, x)</b>	Replace matches in x with a string.
<b>toupper(x)</b>	Convert to uppercase.
<b>tolower(x)</b>	Convert to lowercase.
<b>nchar(x)</b>	Number of characters in a string.

## Factors

<b>factor(x)</b>	<b>cut(x, breaks = 4)</b>
Turn a vector into a factor. Can set the levels of the factor and the order.	Turn a numeric vector into a factor by 'cutting' into sections.

## Statistics

<b>lm(y ~ x, data=df)</b> Linear model.	<b>t.test(x, y)</b> Perform a t-test for difference between means.	<b>prop.test</b> Test for a difference between proportions.
<b>glm(y ~ x, data=df)</b> Generalised linear model.	<b>pairwise.t.test</b> Perform a t-test for paired data.	<b>aov</b> Analysis of variance.
<b>summary</b> Get more detailed information out a model.		

## Distributions

Random Variables	Density Function	Cumulative Distribution	Quantile
Normal	<b>rnorm</b>	<b>dnorm</b>	<b>pnorm</b>
Poisson	<b>rpois</b>	<b>dpois</b>	<b>ppois</b>
Binomial	<b>rbinom</b>	<b>dbinom</b>	<b>pbinom</b>
Uniform	<b>runif</b>	<b>dunif</b>	<b>punif</b>

## Plotting

Also see the **ggplot2** package.

<b>plot(x)</b> Values of x in order.	<b>plot(x, y)</b> Values of x against y.	<b>hist(x)</b> Histogram of x.
---	---	-----------------------------------

## Dates

See the **lubridate** package.