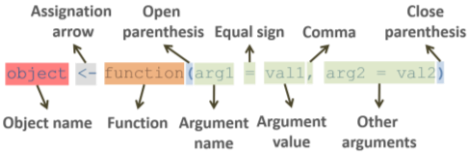
Cheat Sheet

R BASICS WORKSHOP

http://rbasicsworkshop.weebly.com/
30-March-2015
jsebastiantello@gmail.com;
ivan.jimenez@mobot.org

Functions & Arguments

Basic command structure:




```
object <-
    function(argument1=value1,
    argument2=value2)
or
object <- function(value1,
    value2)
or
object <-
    function(argtument2=value2,
    argument1=value1)
```

Help and information:

`?function.name` – open help for function "function.name"
`help(topic)` – open help for a function
www.rseek.org – useful webpage to make R-related searches

Packages:

`install.packages(pkgs)` – install packages
`library(package)` – open installed packages
`installed.packages()` – find details of installed packages
`old.packages()` – find packages with a later available version on the repositories
`update.packages()` – update old packages
`data()` – loads datasets

Objects

Main classes:

Vector (numeric, character, logical) – one dimensional sequence of values
Factor – a variable with "levels" or "categories"
Matrix – a 2-dimensional object
Array – n-dimensional object
Data frame – rows are observations, columns variables of any type
List – object where each element can be of any size or class

Special values:

`NA`, `Inf`, `-Inf`, `NaN`, `NULL`

Value assignation:

```
obj <- val
value -> obj
obj = val
assign(x, value)
```

Object creation:

`numeric(length=0)` – create object of class "numeric"
`character(length=0)` – create object of class "character"
`matrix(data=NA, nrow=1, ncol=1, byrow=FALSE)` – create a matrix
`list()` – create a list

Main object properties:

`ls()` – list all objects in current R session
`rm()` – remove objects from session
`class(x)` – obtain class of x
`mode(x)` – obtain mode of data in x
`names(x)` – obtain element names of x
`rownames(x)`, `colnames(x)` – obtain row or column names of x
`length(x)` – obtain length of x
`dim(x)` – obtain dimensions of x
`nrow(x)`, `ncol(x)` – obtain number of rows or columns in x
`str(object)` – obtain structure of object
`summary(object)` – produce a summary
`table(x)` – calculate a frequency table for values in x

Opening & Saving Data

`getwd()` – return the filepath of current working directory
`setwd(dir)` – set the working directory
`read.table(file, header=FALSE, sep="")` – read a file in table format
`write.table(x, file="", sep=" ", row.names=TRUE, col.names=TRUE)` – save x to a file in table format
`save(…)` – write an R object to a file
`load(file)` – reload datasets written with the function save
`source(file)` – input information from a file (often a script)
`file.choose()` – open window to search for a file

Data Generation

Aggregating data:

`c(…)` – combine values
`paste(…, sep=" ", collapse=NULL)` – concatenate characters
`rbind(…)`, `cbind(…)` – combine by rows or columns
`data.frame(…)` – combine variables into data frame
`merge(x, y)` – merge two data frames
`union(x, y)` – perform set union
`intersect(x, y)` – perform set intersection
`setdiff(x, y)` – perform set asymmetric difference

Sequences:

`:` – generate a regular sequence from x to y
`seq(from=1, to=1, by=((to - from)/(length.out - 1)), length.out=NULL)` – generate a regular sequence
`rep(x, times, each)` – replicate the values in x
`expand.grid(…)` – create a data frame from all **combinations** of the supplied vectors

Data from statistical distributions:

`rnorm(n, mean=0, sd=1)` – generate n random values from a normal distribution
`rpois(n, lambda)` – from a Poisson distribution
`runif(n, min=0, max=1)` – from a uniform distribution
`rbinom(n, size, prob)` – from a binomial distribution

Sampling:

`sample(x, size, replace=FALSE, prob=NULL)` – sample elements of x

Operators

Arithmetic:

`+`, `-`, `*`, `/`, `^` – basic arithmetic operators
`%%` – returns the remainder of x/y
`%/%` – discards remainder of x/y

Relational:

`==` – is x equal to y?
`!=` – not equal to
`>` – greater than
`>=` – greater or equal than
`<` – less than
`<=` – less of equal than

`|` – element-wise or
`&` – element-wise and

Managing Objects

Numeric indexing:

`vector[n]` – return elements "n" of "vector"
`vector[-n]` – return "vector" without "n" elements
`matrix[n]` – return elements "n" of "matrix"
`matrix[row.n, col.n]` – return rows "row.n" and columns "col.n"
`matrix[, col.n]` – return all rows and columns "col.n"
`data.frame[row.n, col.n]` – return rows "row.n" and columns "col.n" of "data.frame"
`data.frame[, col.n]` – return all rows and columns "col.n"
`list[n]` – return elements "n" of "list" in a list format
`list[[n]]` – return concatenated elements "n" of "list"

Logical indexing:

`vector[c(FALSE, TRUE, FALSE)]` – return elements for which condition is TRUE; ***same type of indexing applies to other object classes***

Indexing by name:

`vector["elem.name"]` – return element named "elem.name"; ***same indexing applies to other object classes***

`data.frame$var.name` – returns variable named "var.name"; this ***cannot be applied to matrix*** columns

Other useful functions:

`is.na(x)` – is this an NA?

`!is.na(x)` – is this not an NA?

`na.omit(object)` – eliminate NAs

`which(x)` – identify which elements in x are TRUE

`sort(x, decreasing=FALSE)` – sort vector or factor x

`order(…, decreasing=FALSE)` – return a permutation which rearranges the first argument

`match(x, table)` – return a vector of the positions of matches of the first argument in the second

`t(x)` – transpose x

`diag(x)` – extract the diagonal of matrix x

`lower.tri(x), upper.tri(x)` – return a logical matrix with TRUEs in the lower/upper triangle

`unique(x)` – remove duplicate elements/rows

Statistics

Summary statistics:

`mean(x, na.rm=FALSE)` – calculate arithmetic mean of x

`median(x, na.rm=FALSE)` – median of x

`sd(x, na.rm=FALSE)` – standard deviation of x

`quantile(x, probs=seq(0, 1, 0.25), na.rm=FALSE)` – sample quantiles corresponding to the given probabilities

`range(…, na.rm=FALSE)` – min. and max. values

`min(…, na.rm=FALSE)` – minimum value

`max(…, na.rm=FALSE)` – maximum value

`sum(…, na.rm=FALSE)` – sum of all values in arguments

`rowSums(x, na.rm=FALSE)` – sums of values in each row

`colSums(x, na.rm=FALSE)`

`rowMeans (x, na.rm=FALSE)` – means of values in each row

`colMeans (x, na.rm=FALSE)`

Variable transformations:

`log(x, base=exp(1))` – calculate logarithms of x

`exp(x)` – exponentials

`sqrt(x)` – square roots

`rank(x, na.last=TRUE, ties.method="average")` – rank values of x

`scale(x, center=TRUE, scale=TRUE)` – center and/or standardize x

`round(x, digits=0)` – round x

`ceiling(x)` – round x to the next higher integer (e.g. 3.3 to 4)

`floor(x)` – round x to the next lower integer (e.g. 3.7 to 3)

`cumsum(x)` – return a vector whose elements are the cumulative sums of x

`cumprod(x)` – return a vector whose elements are the cumulative products of x

Basic analyses:

`cor(x, y=NULL, use="everything", method="pearson")` – calculate correlation between x and y, or between pairs of variables in x if a matrix or data frame

`cov(x, y=NULL)` – calculate covariance between x and y, or between pairs of variables in x

`aov(formula, data)` – run an analysis of variance

`lm(formula, data)` – fit a linear model

`glm(formula, family=gaussian, data)` – fit a generalized linear model

`anova(object)` – computes an analysis of variance or deviance for a fitted model

Graphics

High-level functions:

`plot(x, y)` – This is a generic function for multiple types of plots. More frequently, a scatterplot of y against x

`barplot(height)` – a bar-plot where bars come from argument height

`boxplot(x)` – a boxplot of values in x

`hist(x, breaks="Sturges")` – make a histogram of x

`pie(x)` – a pie plot

`pairs(x)` – a matrix of scatterplots

Low-level functions:

`points(x, y)` – add points to a figure

`lines(x, y=NULL)` – lines

`arrows(x0, y0, x1=x0, y1=y0, length=0.25, angle=30)` – arrows

`abline(a=NULL, b=NULL)` – a line based on intercept and slope

`polygon(x, y)` – a polygon

`rect(xleft, ybottom, xright, ytop)` – a rectangle

`text(x, y=NULL, labels=seq_along(x))` – text

`legend(x, y=NULL, legend)` – a figure legend

`axis(side, at=NULL, labels=TRUE)` – an axis

Graphic devices and saving figures:

`jpeg(filename="Rplot%03d.jpeg", width=480, height=480, pointsize=12, quality=75, res=NA)` – open a .jpeg graphic device to save a figure

`pdf(file="Rplots.pdf", width=7, height=7, pointsize=12)` – open a .pdf graphic device to save a figure

`dev.off()` – close a graphic device (saving a file)

`layout(mat) – divide figure into panels`

`par() – set graphical parameters`

`dev.new()` – open a new figure window on the screen

Common arguments for plotting functions:

`pch` – type of symbol in scatterplots

`lty` – type of line

`col` – color

`bg` – background color

`border` – border color

`lwd` – width of line

`cex` – size of symbol

`cex.lab` – size of axis label

`cex.axis` – size of axis numbering

`xlim, ylim` – limits in x or y dimension

`xlab, ylab` – labels for x or y axis

`axes` – logical indicating whether axes should be plotted

`type` – type of scatterplot

`las` – orientation of numbering in y axis

Flow Control

`try({expression}, silent=FALSE)` – run "expression", if it generates and error, continue running the script (when `silent=TRUE`).

Loops:

`for(var in seq){expression}` – repeat "expression" as many times as there are elements in the vector "seq". At each iteration, "var" takes a value from "seq"

`while(condition){expression}` – repeat "expression" while "condition" is TRUE

Conditions:

`if(condition){expression}` – if "cond" is TRUE, run "expression"

`ifelse(test, yes, no)` – if "test" is TRUE, run "yes", otherwise run "no"

Breaks:

`next` – halt the processing of the current iteration and advance the looping index

`break` – break out of a loop

`stop()` – stop execution of the current expression and execute an error action