StudyR

PX

3/5/2020

Variables

```r
# removing variable

a <- 2

rm(a)

# variable names are case sensitive
```

Data Types

```r
# how to check ?
x <- 2
class(x)
```

```
## [1] "numeric"
```

```r
# numeric data, how to check?

as.numeric(x)
```

```
## [1] 2
```

```r
# to check integer or not, setup an integer with L

b <- 24L
b
```

```
## [1] 24
```

```r
is.integer(b)
```

```
## [1] TRUE
```

```r
# character data, two types in R: character and factor

y <- "Houston"
y
```

```
## [1] "Houston"
```

```r
yy <- factor("Houston")
yy
```

```
## [1] Houston
## Levels: Houston
```

```
# character variable is also case sensitive,
# to check the length of the data, use the command nchar function

nchar(x)
```

```
## [1] 1
```

```
nchar(y)
```

```
## [1] 7
```

```
# Will generate one error if run the following code
# nchar(yy)


# Dates
date1 <- as.Date("2020-03-05")
date1
```

```
## [1] "2020-03-05"
```

```
class(date1)
```

```
## [1] "Date"
```

```
# logical

FALSE*7
```

```
## [1] 0
```

```
TRUE*11
```

```
## [1] 11
```

```
k <- TRUE
class(k)
```

```
## [1] "logical"
```

```
"data" == "date"
```

```
## [1] FALSE
```

```
# vectors, big part.
# vector is a collection of elements.
# A vector can NOT be of mixed type.
# Vectors do NOT have a dimension.

xx <- c(1, 2, 3)
xx
```

```
## [1] 1 2 3
```

```
class(xx)
```

```
## [1] "numeric"
```

```
# vector operation
xx*7
```

```
## [1]  7 14 21
```

```
xx+2
```

```
## [1] 3 4 5
```

```
xx/3
```

```
## [1] 0.3333333 0.6666667 1.0000000
```

```
xx^3
```

```
## [1]  1  8 27
```

```
sqrt(xx)
```

```
## [1] 1.000000 1.414214 1.732051
```

```
1:10
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
10:1
```

```
##  [1] 10  9  8  7  6  5  4  3  2  1
```

```
-7:9
```

```
##  [1] -7 -6 -5 -4 -3 -2 -1  0  1  2  3  4  5  6  7  8  9
```

```
5:-24
```

```
##  [1]   5   4   3   2   1   0  -1  -2  -3  -4  -5  -6  -7  -8  -9 -10 -11 -12 -13
## [20] -14 -15 -16 -17 -18 -19 -20 -21 -22 -23 -24
```

```
xxx <- 1:10
yyy <- -5:4

xxx+yyy
```

```
##  [1] -4 -2  0  2  4  6  8 10 12 14
```

```
xxx^yyy
```

```
##  [1] 1.000000e+00 6.250000e-02 3.703704e-02 6.250000e-02 2.000000e-01
##  [6] 1.000000e+00 7.000000e+00 6.400000e+01 7.290000e+02 1.000000e+04
```

```
# Thing becomes a little bit complicated for
# the operation between factors with two different length
# The shorter vector get recycled

xxx+c(1, 2)
```

```
##  [1]  2  4  4  6  6  8  8 10 10 12
```

```
# this will give warnning, since one is not multiple of the other
yyy +c(1,2,3)
```

```
## Warning in yyy + c(1, 2, 3): longer object length is not a multiple of shorter
## object length
```

```
##  [1] -4 -2  0 -1  1  3  2  4  6  5
```

```
# comparison between vectors

xxx >= 5
```

```
##  [1] FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```r
xxx < yyy
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```r
# test for all of vectors
any(xxx < yyy)
```

```
## [1] FALSE
```

```r
# nchar will also perform on vectors

# accessing elements in vector will use []
xxx[1]
```

```
## [1] 1
```

```r
xxx[1:5]
```

```
## [1] 1 2 3 4 5
```

```r
xxx[c(1,7)]
```

```
## [1] 1 7
```

```r
# name a vector
w <- 1:3
names(w) <- c("a","b","c")
w
```

```
## a b c
## 1 2 3
```

```r
# factors are important concept for R language.
q <- c("basketball", "scooer", "valleyball", " pingpang", "tennis","basketball")

q2factor <- as.factor(q)
q2factor
```

```
## [1] basketball scooer     valleyball  pingpang  tennis     basketball
## Levels:  pingpang basketball scooer tennis valleyball
```

```r
# notice that after printing q2factor, there is a level part.
# The levels of a factor are the unique values of that factor varialbe.
# R is giving each unique value of a factor a unique INTEGER tying
# it back to the character representation.
as.numeric(q2factor)
```

```
## [1] 2 3 5 1 4 2
```

```r
# Normally, the order of levels does not matter
# one level is no different from another.
# However, if order matters, we do the following
p <- factor(c("BS", "MS", "PHD"), levels = c("BS", "MS", "PHD"), ordered = TRUE)
p
```

```
## [1] BS  MS  PHD
## Levels: BS < MS < PHD
```

```r
# function calling
mean(xxx)
```

```
## [1] 5.5
```

```
# to find more details of a function
?mean

# to find more by part of the names
apropos("mea")
```

```
##  [1] ".colMeans"          ".rowMeans"          "colMeans"
##  [4] "influence.measures" "kmeans"             "mean"
##  [7] "mean.Date"          "mean.default"       "mean.difftime"
## [10] "mean.POSIXct"       "mean.POSIXlt"       "rowMeans"
## [13] "weighted.mean"
```

```
# Missing data: two types in R
# NA and NULL

# missing data NA is part of the vector
 z <- c( 1, 3, NA, 4, 9)

 # check if is.na
 is.na(z)
```

```
## [1] FALSE FALSE  TRUE FALSE FALSE
```

```
 # NULL is nothingness. And NULL will not be stored in a vector

 zz <- c(1, NULL, 3)
 zz
```

```
## [1] 1 3
```

```
 is.null(zz)
```

```
## [1] FALSE
```

Advanced Data Structures

```
# Data.frame is one of the most used structures in R

x <- 1:3
y <- 4:6
q <- c("ba", "ma", "xi")
family <- data.frame(x,y,q)
family
```

```
##   x y  q
## 1 1 4 ba
## 2 2 5 ma
## 3 3 6 xi
```

```
class(family)
```

```
## [1] "data.frame"
```

```
# assign colume names
family <-data.frame( First = x, Second = y, Love = q)
family
```

```
##   First Second Love
## 1     1      4   ba
## 2     2      5   ma
## 3     3      6   xi
```

```r
# check the length, names
nrow(family)
```

```
## [1] 3
```

```r
ncol(family)
```

```
## [1] 3
```

```r
dim(family)
```

```
## [1] 3 3
```

```r
# this will help us check colume names
names(family)
```

```
## [1] "First"  "Second" "Love"
```

```r
names(family)[2]
```

```
## [1] "Second"
```

```r
# check row names and assign row names
rownames(family)
```

```
## [1] "1" "2" "3"
```

```r
rownames(family) <- c("one", "two" , "three")

rownames(family)
```

```
## [1] "one"   "two"   "three"
```

```r
# back to default
rownames(family) <- NULL
rownames(family)
```

```
## [1] "1" "2" "3"
```

```r
# check the first or last few rows
head(family)
```

```
##   First Second Love
## 1     1      4   ba
## 2     2      5   ma
## 3     3      6   xi
```

```r
tail(family)
```

```
##   First Second Love
## 1     1      4   ba
## 2     2      5   ma
## 3     3      6   xi
```

```r
# access the column or sepcific data in the data.frame
family$Love
```

```
## [1] ba ma xi
## Levels: ba ma xi
```

```
family[2,1]
```

```
## [1] 2
```

```
family[2,1:2]
```

```
##   First Second
## 2     2      5
```

```
# first and third row, column 2 to 3
family[c(1,3),2:3]
```

```
##   Second Love
## 1      4   ba
## 3      6   xi
```

```
# all of column 3
family[,3]
```

```
## [1] ba ma xi
## Levels: ba ma xi
```

```
family[, 2:3]
```

```
##   Second Love
## 1      4   ba
## 2      5   ma
## 3      6   xi
```

```
family[2,]
```

```
##   First Second Love
## 2     2      5   ma
```

```
family[,c("Love")]
```

```
## [1] ba ma xi
## Levels: ba ma xi
```

```
# to access one column, it can return as a factor, or a data.frame column

family[,"Love"]
```

```
## [1] ba ma xi
## Levels: ba ma xi
```

```
class(family[,"Love"])
```

```
## [1] "factor"
```

```
family["Love"]
```

```
##   Love
## 1   ba
## 2   ma
## 3   xi
```

```
class(family["Love"])
```

```
## [1] "data.frame"
```

```r
family[["Love"]]
```

```
## [1] ba ma xi
## Levels: ba ma xi
```

```r
class(family[["Love"]])
```

```
## [1] "factor"
```

```r
# to maintain as a single column data.frame using single brackets, add drop=FALSE
family[,"Love",drop=FALSE]
```

```
##   Love
## 1   ba
## 2   ma
## 3   xi
```

```r
class(family[,"Love",drop=FALSE])
```

```
## [1] "data.frame"
```

```r
family[,3,drop=FALSE]
```

```
##   Love
## 1   ba
## 2   ma
## 3   xi
```

```r
class(family[,3,drop=FALSE])
```

```
## [1] "data.frame"
```

```r
# to see how factors are represented in data.frame form,
# use model.matrix to create a set of indicator.
# That is one column for each level of a factor,
# with a 1 if a row contains that level or 0 otherwise.

newFactor <-factor(c("aa","bb","cc","dd","aa","dd"))
model.matrix(~newFactor - 1)
```

```
##   newFactoraa newFactorbb newFactorcc newFactordd
## 1           1           0           0           0
## 2           0           1           0           0
## 3           0           0           1           0
## 4           0           0           0           1
## 5           1           0           0           0
## 6           0           0           0           1
## attr(,"assign")
## [1] 1 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$newFactor
## [1] "contr.treatment"
```

```r
# List : will hold arbitrary objects of either same type or varying types.
# store any number of items of any type.

# create a three element list in memory of KB
```

```r
list(2,8,24)
```

```
## [[1]]
## [1] 2
## 
## [[2]]
## [1] 8
## 
## [[3]]
## [1] 24
```

```r
# create a single element is a vector(has three elements)
list(c(2,8,24))
```

```
## [[1]]
## [1]  2  8 24
```

```r
# two element list

list1 <- list(c(2,8,24), 1996:2016)
list1
```

```
## [[1]]
## [1]  2  8 24
## 
## [[2]]
##  [1] 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
## [16] 2011 2012 2013 2014 2015 2016
```

```r
list2 <- list(family, 2013: 2113)
list2
```

```
## [[1]]
##   First Second Love
## 1     1      4   ba
## 2     2      5   ma
## 3     3      6   xi
## 
## [[2]]
##   [1] 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027
##  [16] 2028 2029 2030 2031 2032 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042
##  [31] 2043 2044 2045 2046 2047 2048 2049 2050 2051 2052 2053 2054 2055 2056 2057
##  [46] 2058 2059 2060 2061 2062 2063 2064 2065 2066 2067 2068 2069 2070 2071 2072
##  [61] 2073 2074 2075 2076 2077 2078 2079 2080 2081 2082 2083 2084 2085 2086 2087
##  [76] 2088 2089 2090 2091 2092 2093 2094 2095 2096 2097 2098 2099 2100 2101 2102
##  [91] 2103 2104 2105 2106 2107 2108 2109 2110 2111 2112 2113
```

```r
# name in a list

names(list2) <- c("My Love", "100 years")

list2
```

```
## $`My Love`
##   First Second Love
## 1     1      4   ba
## 2     2      5   ma
## 3     3      6    xi
##
## $`100 years`
##   [1] 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027
##  [16] 2028 2029 2030 2031 2032 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042
##  [31] 2043 2044 2045 2046 2047 2048 2049 2050 2051 2052 2053 2054 2055 2056 2057
##  [46] 2058 2059 2060 2061 2062 2063 2064 2065 2066 2067 2068 2069 2070 2071 2072
##  [61] 2073 2074 2075 2076 2077 2078 2079 2080 2081 2082 2083 2084 2085 2086 2087
##  [76] 2088 2089 2090 2091 2092 2093 2094 2095 2096 2097 2098 2099 2100 2101 2102
##  [91] 2103 2104 2105 2106 2107 2108 2109 2110 2111 2112 2113
```

```r
#

# create an empty list of a certain size, using vector
(emptyList <- vector(mode = "list", length = 7))
```

```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
##
## [[7]]
## NULL
```

```r
# to access element in a list,
# use double bracket, specifying either number or name

list2[[1]]
```

```
##   First Second Love
## 1     1      4   ba
## 2     2      5   ma
## 3     3      6   xi
```

```r
list2[["My Love"]]
```

```
##   First Second Love
## 1     1      4   ba
## 2     2      5   ma
## 3     3      6   xi
```

```r
# access elements of element, nested index will be used
list2[[1]]$Love
```

```
## [1] ba ma xi
## Levels: ba ma xi
```

```
# can append elements to list
length(list2)
```

```
## [1] 2
```

```
list2[[3]] <- c("LOVELOVELOVE")
length(list2)
```

```
## [1] 3
```

```
# add new elements

list2[["Newmember"]] <- 1:99

names(list2)
```

```
## [1] "My Love"   "100 years" ""          "Newmember"
```

```
# Matrices

A <- matrix(1:9, nrow = 3, ncol = 3)
A
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
```

```
A[-1,] # select all rows except first
```

```
##      [,1] [,2] [,3]
## [1,]    2    5    8
## [2,]    3    6    9
```

```
# matrix multiplication
A %*% A
```

```
##      [,1] [,2] [,3]
## [1,]   30   66  102
## [2,]   36   81  126
## [3,]   42   96  150
```

```
# names
colnames(A)
```

```
## NULL
```

```
rownames(A)
```

```
## NULL
```

```
# Array : multidimensional vector
# first element in c is row index
# second is column index
# third is outer dimension

theArray <- array(1:12, dim=c(2,3,2))
theArray
```

```
## , , 1
##
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
##
## , , 2
##
##      [,1] [,2] [,3]
## [1,]    7    9   11
## [2,]    8   10   12
```

```
theArray[1, , ]
```

```
##      [,1] [,2]
## [1,]    1    7
## [2,]    3    9
## [3,]    5   11
```

```
theArray[1, , 1]
```

```
## [1] 1 3 5
```

```
theArray[, , 2]
```

```
##      [,1] [,2] [,3]
## [1,]    7    9   11
## [2,]    8   10   12
```

```
# The key difference between matrix and array
# is that matrices are only two dimensions,
# while arrays can have any dimensions.
```